

**RADC-TR-83-184**

**Phase Report**

**March 1984**



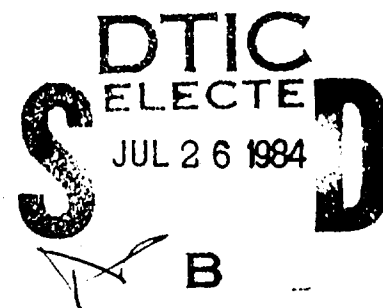
12

# ***SOFTWARE SUPPORT FOR A VIDEO DISC BASED TERRAIN MAP DISPLAY SYSTEM***

**The MITRE Corporation**

**evin Ilse**

**APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED**



**ROME AIR DEVELOPMENT CENTER  
Air Force Systems Command  
Griffiss Air Force Base, NY 13441**

84 07 26 019

AD-A143 700

DTIC FILE COPY

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-83-184 has been reviewed and is approved for publication.

APPROVED:



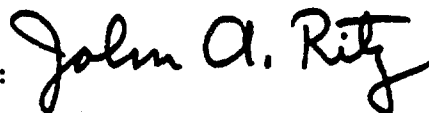
FREDERICK W. RAHRIG  
Project Engineer

APPROVED:



THADEUS J. DOMURAT  
Acting Technical Director  
Intelligence & Reconnaissance Division

FOR THE COMMANDER:



JOHN A. RITZ  
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC ( IRRE ) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-83-184	2. GOVT ACCESSION NO. <b>A143700</b>	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SOFTWARE SUPPORT FOR A VIDEO DISC BASED TERRAIN MAP DISPLAY SYSTEM		5. TYPE OF REPORT & PERIOD COVERED Phase Report March 1984
		6. PERFORMING ORG. REPORT NUMBER MTR8828
7. AUTHOR(s) Kevin Ilsen		8. CONTRACT OR GRANT NUMBER(s) F19628-82-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS The MITRE Corporation Burlington Road Bedford MA 01730		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS MOIE7330
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (IRRE) Griffiss AFB NY 13441		12. REPORT DATE March 1984
		13. NUMBER OF PAGES 116
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE N/A
16. DISTRIBUTION STATEMENT (of this Report)  Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)  Same		
18. SUPPLEMENTARY NOTES RADC Project Engineer: Frederick W. Rahrig (IRRE)		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Video Disc Map Display		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report describes the utility of video disc technology for storing, access- ing and viewing terrain map data for various applications. This document also addresses the hardware configuration and software support required to achieve the objective of a softcopy map display architecture.		

# ABSTRACT

The large capacity and random access capability of optical scan video discs make them an excellent storage medium for background maps used in tactical situation displays. Early experience with paper map photographs on video discs revealed serious resolution and clutter problems when these maps are overlaid with intelligence symbology. To improve the displays, the Video Frame Store (VFS) was built. The VFS can load a series of video disc frames, each containing an image of an individual feature, which together form a composite map of the area of interest. Once the images have been loaded, they can be manipulated digitally by microcode programs in the VFS.

This paper describes the support software which was written to turn the video disc and the VFS into a full-scale terrain map display system. Three interactive software packages are described in detail, with full user manuals in the appendices. The data base structure for map applications is described, as well as the video disc layout. Also included are projections of short term and long term plans to continue these efforts, and a general survey of commercially available devices with features comparable to the Video Frame Store.

**DTIC**  
**ELECTE**  
**S** JUL 26 1984 **D**  
**B**

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

## ACKNOWLEDGEMENTS

The author wishes to thank the following people for their help in the preparation of this report: Deborah Centola, for typing and formatting assistance; Sandra Bonnell, for proofreading the entire document; John White, for insightful editorial comments; Linda Morrill, for putting all the pieces together; and David Lehman, for general input to the content and for overall project supervision.

# DISCLAIMER

The use of generic masculine pronouns has been adopted in text references to individuals whose gender is not otherwise established. This has been done solely for succinctness of expression and such references are intended to apply equally to men and women.

## TABLE OF CONTENTS

<u>Section</u>		<u>Page</u>
1	INTRODUCTION	1
1.1	MAP DISPLAY EVOLUTION	1
1.2	VIDEO FRAME STORE ARCHITECTURE	2
1.3	VFS SYSTEM OBJECTIVE	3
2	VFS INTERFACE APPROACH	4
2.1	THE UNKNOWN USER	4
2.2	AN OPEN-ENDED INTERFACE	4
2.3	SOFTWARE USER DEMOGRAPHY	6
2.4	SOFTWARE PACKAGES NEEDED	6
3	THE VFS EXECUTIVE	8
3.1	BASIC FEATURES OF THE EXECUTIVE	8
3.2	VFS EXECUTION COMMANDS	9
	3.2.1 RUN Command	9
	3.2.2 HALT Command	9
	3.2.3 RESET Command	10
3.3	MEMORY ALTERATION	10
	3.3.1 SET Command	10
	3.3.2 CLEAR Command	10
	3.3.3 DRAW Command	10
	3.3.4 ERASE Command	10
3.4	DUMPING DATA	10
	3.4.1 DUMP Command	11
	3.4.2 LOAD Command	11
3.5	THE VFS INTERACTIVE DEBUGGER	11
4	APPLICATION DATABASE STRUCTURE	13
4.1	CONCEPTUAL ORGANIZATION	13
4.2	VIDEO DISC LAYOUT	15

## TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Page</u>
4.3 DATA FILES	19
4.3.1 The Video Disc Directory	19
4.3.2 The Feature Color Table	22
4.3.3 The Application Definition File	24
5 APPLICATION DEFINITION	28
5.1 OVERVIEW	28
5.2 DATA ENTRY	28
5.2.1 Number of Levels	29
5.2.2 Video Disc Directory "Entry"	29
5.2.3 Video Disc Directory "Sub-entry"	29
5.2.4 Number of Features	29
5.2.5 Features and Colors	30
5.2.6 Feature Intersections	30
5.2.7 Application Definition File Constants	30
5.2.8 Black and White Background Frames	30
5.2.9 Foreground Frames	30
5.3 THE DATABASE REPORT	31
6 INTERACTIVE APPLICATION CONTROL	32
6.1 INVOCATION FORMS	32
6.2 FEATURE DISPLAY CONTROL	33
6.2.1 ADD Command	33
6.2.2 ERASE Command	33
6.2.3 REMOVE Command	33
6.3 COLOR AND PRECEDENCE CONTROL	33
6.3.1 COLOR Command	33
6.3.2 COLOR Command for Intersections	34
6.3.3 PRECEDE Command	34
6.4 PICTURE MANIPULATION	34
6.4.1 ZOOM Command	34
6.4.2 TRANSLATE Command	34



## TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Page</u>
6.4.3 SLIDE Command	34
6.4.4 JOYSTICK Command	34
6.5 VIDEO DISC INTERFACE	35
6.5.1 DISC Command	35
6.5.2 SEARCH Command	35
6.6 MISCELLANEOUS	35
6.6.1 QUERY Command	35
6.6.2 LOAD Command	36
6.6.3 SYNC Command	36
6.6.4 PAUSE Command	36
7 INDIRECT APPLICATION CONTROL	37
8 INITIAL DEMONSTRATION VIDEO DISC	39
8.1 GEOGRAPHIC AREA COVERED	39
8.2 MAP FEATURES	41
9 SUMMARY AND FUTURE DIRECTIONS	44
9.1 SUMMARY OF ACCOMPLISHMENTS	44
9.2 FUTURE WORK	44
9.3 LONG-RANGE IMPROVEMENTS	46
9.3.1 Larger Image Memory	46
9.3.2 Multiple Video Discs	47
9.3.3 Transparent Microcode Swapping	48
9.3.4 User Annotation of Maps	48
9.4 SURVEY OF COMMERCIALY AVAILABLE SYSTEMS	49
9.4.1 Requirements of an Off-the-Shelf Device	49
9.4.2 Image Processors	51
9.4.3 Graphic Display Controllers/Terminals	52
9.4.4 Modular Boards	54
LIST OF REFERENCES	57

## TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Page</u>
APPENDIX A    VFS EXECUTIVE USER'S MANUAL	59
A.1        INTRODUCTION	59
A.2        BASICS	59
A.2.1   Invoking the VFS Executive	59
A.2.2   Line and Loop Commands	59
A.2.3   Exiting	60
A.3        COMMAND ARGUMENTS	60
A.3.1   Syntax	60
A.3.2   Memory Name Arguments	61
A.3.3   Memory Specification Switches	61
A.3.4   Global Switches	62
A.4        INDIRECT COMMAND FILES	62
A.5        UNIX COMMAND EXECUTION	63
A.6        COMMANDS	63
A.6.1   HELP Command	64
A.6.2   PAUSE Command	64
A.6.3   RUN Command	64
A.6.4   HALT Command	64
A.6.5   RESET Command	64
A.6.6   CLEAR Command	65
A.6.7   SET Command	65
A.6.8   LOAD Command	65
A.6.9   DUMP Command	65
A.6.10   DRAW Command	66
A.6.11   ERASE Command	66
A.7        THE VFS INTERACTIVE DEBUGGER (VID)	67
A.7.1   Arguments	68
A.7.2   Break Characters	68
A.7.2.1   "/"	70
A.7.2.2   "RETURN"	70
A.7.2.3   "LINE FEED"	70
A.7.2.4   "^"	70
A.7.2.5   "N"	70

# TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Page</u>
A.7.2.6 "P"	70
A.7.2.7 "L"	70
A.7.2.8 "E"	70
A.7.2.9 "H"	70
A.7.2.10 "U"	70
A.7.2.11 "X"	70
A.7.2.12 "S"	70
A.7.2.13 "Z"	71
A.7.2.14 "Q" or "ctrl-d"	71
A.7.2.15 "F1"	71
A.7.2.16 "?"	71
A.7.3 Indirect Command Files	71
A.7.4 UNIX Command Execution	71
A.8 EXECUTIVE ERROR MESSAGES	71
A.8.1 Top Level Command Entry	72
A.8.2 Command Input Loop Arguments	72
A.8.3 HELP Command Topic Name	72
A.8.4 RUN Command Transfer Address	73
A.8.5 CLRAR and SET Command Line Arguments	73
A.8.6 LOAD Command Line Arguments	73
A.8.7 DUMP Command Line Arguments	74
A.8.8 VID Messages	74
APPENDIX B APPLICATION DEFINITION USER'S MANUAL	77
B.1 INTRODUCTION	77
B.2 BASICS	77
B.2.1 Definition - Pass 1	77
B.2.2 Definition - Pass 2	78
B.2.3 Database Report	79
B.3 DEFINING THE VIDEO DISC DIRECTORY	79
B.3.1 Map Scale Information	79
B.3.2 Photo Depth	80
B.3.3 Photo Size	80
B.3.4 Strip Latitudes	80
B.3.5 Strip Input Loop	80

## TABLE OF CONTENTS (Continued)

<u>Section</u>		<u>Page</u>
B.4	DEFINING THE FEATURE COLOR TABLE	82
B.4.1	Feature Name	82
B.4.2	Feature Color	83
B.4.3	Feature Intersections	83
B.4.3.1	Intersection Color	84
B.4.3.2	Strong or Weak	84
B.4.3.3	Feature List	84
B.5	DEFINING THE APPLICATION DEFINITION FILE	85
B.5.1	Intensity Mapping Information	85
B.5.2	Background Control	85
B.5.3	Black and White Background Frames	86
B.5.3.1	Write Masks	86
B.5.3.2	A/D Mode	87
B.5.3.3	A/D Limits	87
B.5.4	Foreground Frames	87
B.5.4.1	Feature Name	87
B.5.4.2	Feature Type	87
B.5.4.3	Bitplane Number	88
B.5.4.4	A/D Mode	88
B.5.4.5	A/D Limits	88
B.6	DATA ENTRY ERRORS	89
APPENDIX C	APPLICATION CONTROLLER USER'S MANUAL	91
C.1	INTRODUCTION	91
C.2	BASICS	91
C.2.1	Using the Video Disc	91
C.2.2	Using a Pre-Loaded Image	91
C.2.3	Loading a Digital Image File	92
C.3	COMMAND SYNTAX	92
C.4	INDIRECT COMMAND FILES	92
C.5	UNIX COMMAND EXECUTION	93

# TABLE OF CONTENTS (Concluded)

<u>Section</u>	<u>Page</u>
C.6      COMMANDS	93
C.6.1    Feature Selection	94
C.6.1.1    ADD	94
C.6.1.2    ERASE	94
C.6.1.3    REMOVE	94
C.6.2    Color Control	94
C.6.2.1    COLOR	94
C.6.2.2    COLOR for Intersections	94
C.6.2.3    PRECEDE	95
C.6.3    Picture Manipulation	95
C.6.3.1    ZOOM	95
C.6.3.2    TRANSLATE	95
C.6.3.3    SLIDE	96
C.6.3.4    JOYSTICK	96
C.6.4    Video Disc Control	96
C.6.4.1    DISC	96
C.6.4.2    SEARCH	96
C.6.5    Miscellaneous	96
C.6.5.1    QUERY	96
C.6.5.2    LOAD	97
C.6.5.3    SYNC	97
C.6.5.4    PAUSE	97
DISTRIBUTION LIST	99

## SECTION 1

### INTRODUCTION

#### 1.1 MAP DISPLAY EVOLUTION

Tactical situation displays originally took the form of large paper maps, with pertinent locations marked by pins or flags. The coordinates of these locations might even be gathered or transmitted automatically; the military analyst would then read the information from a CRT display or computer printout, and mark the map by hand.

With the development of computer graphics, systems were designed to display geographic maps directly on a CRT with special symbols marking important locations. This unburdened the analyst from the task of managing the placement of the markers, as well as providing a central work station which did not require running back and forth between a CRT and a wall map. However, all of the advantages of the traditional paper map were lost in this scheme. Usually, a computer-displayed map consisted only of major features such as political boundaries, rivers, etc. Other detailed features were not included; features which might be relevant to the analyst's particular application, such as roads, tree cover, aircraft obstructions, etc.

The introduction of the optical video disc presented a new approach to background map display. A video disc can hold up to 54,000 individual full color frames on a single side. Photographs of standard color maps for a particular region can be placed on the disc, at multiple scale levels, with enough room on the disc to photograph the map sections with sufficient overlap to bring virtually any selected point to the center of the screen.

Such a video disc map data base captures all of the features present on a paper map; the analyst sees on the screen what he would see on a map tacked to the wall. (There is, however, an appreciable loss of resolution as compared to a printed paper map.) The analyst can sit before the screen and work with the tactical data overlaid on the map; he never has to get up and run to a paper map on the wall. Unfortunately, this means that the distance from his eyes to the map always stays the same. When the paper map was on the wall, the analyst could change the apparent level of detail by physically stepping back from the map or moving in closer. On the CRT, all of the features are present, and seem to persist even if the analyst

does step back from the screen. The advantage of detail on the paper map has turned into a disadvantage of clutter on the CRT display.

The problem of map clutter can be dealt with most effectively if the analyst has selective control over which features are displayed. He could then tailor his display to his individual application, or to the particular investigation in which he is involved. Such control would require a separate memory plane for each map feature--roads, railroads, rivers, etc.--as well as a database of images of individual map features. Such a digital database for even a small geographic region represents a staggering storage requirement. For example, if there are 16 map features, each 512 line x 512 pixel map would consume 4 megabytes of storage.

The Video Frame Store system exploits the high capacity of the optical video disc by storing map feature separations on a video disc in standard NTSC format. The video disc frames are digitized by programmable A/D convertors and stored in individual memory planes of a 24-bits-deep frame buffer. The frame buffer device includes special purpose hardware for digital processing of the composite map image, permitting easy control over feature display, feature color mapping, and even advanced techniques such as integral and non-integral zoom.

## 1.2 VIDEO FRAME STORE ARCHITECTURE

The Video Frame Store (VFS) is a digital frame buffer and graphics processing device. It consists of a 512-line by 512-pixel image memory, with 24 bits per pixel (8 bitplanes for each of the three primary colors). Additional memories are also provided for microcode programs, intensity transformation, a color lookup table, and image processing convolution coefficients.

Within the 8-bit-per-color image memories, from 4 to 6 bits of each color group may be used for a full-color digitization of "background" color data; the remaining bits are used as indices into a 4096-word color lookup table to produce "foreground" displays. The background color memories are used to load a full color map background. Individual map feature images are loaded into the separate foreground bitplanes.

One component of the VFS, the Video Processing Element, performs digital manipulations on the displayed image. Screen center-point translation can be accomplished instantaneously by setting line and pixel offset registers. The Video Processing Element contains a 2 x 2 pixel convolver which is controlled by the

1024-instruction microprogram memory. The convolver can be programmed to provide both integer and non-integer zooming at the frame rate (30 Hz, interlaced).

The image memory will accept input through programmable A/D convertors from such devices as a video camera, a video cassette recorder, and an optical video disc player.

An IEEE 488 interface provides interaction with a host processor for loading digitally encoded images, downloading to micro memory, intensity mapping memory, coefficient memory, and the color lookup table, and setting various control registers (A/D control, microprogram start address, bitplane display masks, etc.).

A thorough overview of the VFS architecture and features as applied to the terrain map display problem may be found in [1].

### 1.3 VFS SYSTEM OBJECTIVE

The objective of the Video Frame Store is to serve as a Terrain Map Display Sub-system for situation display applications. In such a system, the VFS would provide the background map, and a separate system would overlay intelligence data on the map. Achieving this goal involves the construction of a software or software/hardware front end to the VFS, so that a user or a co-system does not have to be concerned with the details of the VFS architecture.

This report describes all of the application software developed to achieve this goal for the VFS. Future directions in the software development cycle are discussed, as well as possible hardware improvements based on the current state of the art and experience with the VFS. User manuals for interactive software packages appear in the appendices at the end of the report.



## SECTION 2

### VFS INTERFACE APPROACH

#### 2.1 THE UNKNOWN USER

As was mentioned in the introduction, the VFS is being developed to serve as a general purpose sub-system which will provide background map displays. The class of applications for the VFS is thus well defined, but the particular application is not. Specifically, the ultimate user, or users, of the VFS is unknown.

The key ingredient in designing an effective user interface is the user--his needs, his objectives, his degree of familiarity with various input devices and technical jargon. Ideally, a user interface is built upon a strong base of concepts already familiar to the user; the user can then quickly become comfortable with the interface. It is impossible to create a successful user interface when the user is unknown.

#### 2.2 AN OPEN-ENDED INTERFACE

If the VFS is to be a portable sub-system, the responsibility of interfacing with the user will have to lie with the application system, not the VFS software front end. The VFS requires a host computer, and this host will ultimately communicate not with a user, but with another software system. For true portability, this system should not be constrained to being implemented on the VFS host machine. So, the VFS host must be capable of communicating with another processor.

A well designed VFS application front end should make the VFS terrain map display system so portable that it can be added to an existing intelligence display application system with as little impact as possible on the software package of the application. For this to be possible, all VFS control functions must be handled by subroutines which can be invoked through a message-passing interface from another computer.

An example of a dual-processor configuration using the VFS to provide background map display is illustrated in Figure 1.

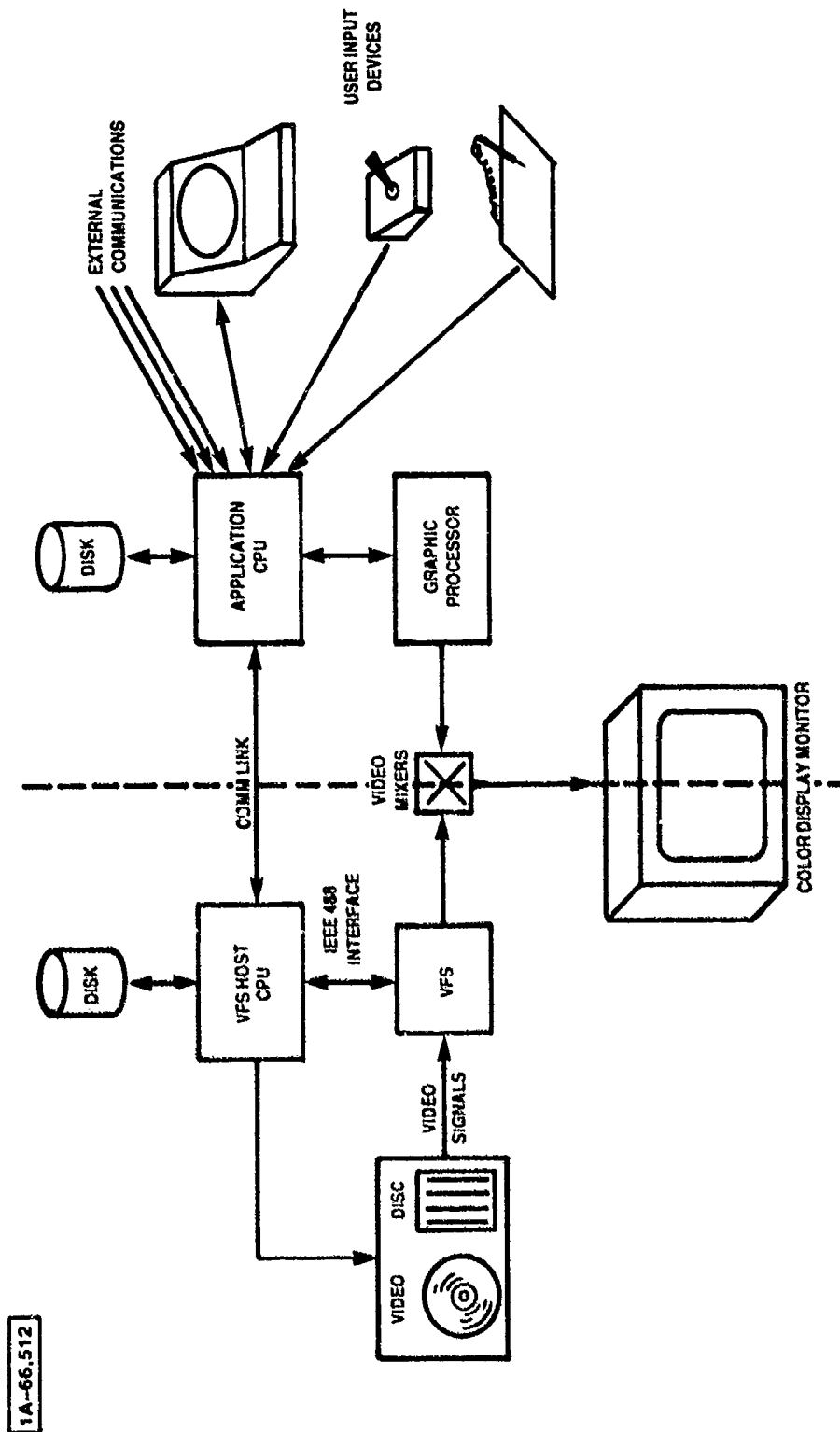


Figure 1. SAMPLE DUAL-PROCESSOR CONFIGURATION

### 2.3 SOFTWARE USER DEMOGRAPHY

Besides the end user, who interfaces indirectly with the VFS through his own application package, there are other types of VFS users who must be accommodated.

Graphics programmers will have to interface directly with the VFS device and must have access to and control over all VFS components. These individuals will be involved in graphic manipulations independent of context--functions such as synchronizing frame digitization with the raster sweep, zooming at various levels of magnification, translating the image, etc. The graphics programmers will also need to experiment with the various VFS memories--color transformation, intensity transformation, and coefficient memory. All of these tasks have nothing to do with terrain map display; they deal only with intensity levels and pixels.

The programmer of an intelligence application system which uses the VFS must develop a database on the VFS host processor to drive the VFS for his particular application. This database includes actual maps on the video disc, a directory of these maps, and a description of how the maps are loaded into the VFS memories. A versatile, easy to use interface is required for database definition.

Finally, the ability to demonstrate the VFS as a terrain map display system is necessary. VFS developers will pose as end users and will need a user interface to manipulate a background map display. A VFS developer demonstrating the system's features differs from a typical end user both in needs and in knowledge. An "idiot-proofed" interface for the naive user, complete with detailed error messages and on-line help features, is not necessary for this task. Conversely, certain functions such as loading files of coefficients will be provided here, while they will be absent in typical end user interfaces. A simple command line dialogue, running on the VFS host processor, will suffice for the interaction.

### 2.4 SOFTWARE PACKAGES NEEDED

The VFS user demography defines a group of software packages to be developed on the host computer. The packages are:

- a VFS Executive, for graphics programming
- an application definition package

- an application controller, for interactive demonstrations
- an interprocessor message handler and subroutine package

These packages are described in sections 3, 5, 6, and 7 of this report. Detailed user manuals are also provided in the appendices.

## SECTION 3

### THE VFS EXECUTIVE

The original host for the VFS was a PDP-11/03 microcomputer, and a VFS Executive program was running on that machine. This machine was adequate to handle the processing requirements of the VFS host. However, to provide a more supportive software development environment, the host was upgraded to an Interdata 8/32 minicomputer running Version 6 UNIX, and the Executive was rewritten for the new operating system with syntax and style preserved, although a number of enhancements were made.

The purpose of the Executive is to allow control over all elements of the VFS device, independent of context. No associations are made with background maps; driven by low level commands in the Executive, the VFS could function equally well as an image processing device, or even a raster display system. Any writable memory location in the VFS may be set from the Executive, the microsequencer can be started or halted, data can be dumped to a file or to the printer, etc. The Executive is thus the primary interface to the VFS.

The basic commands of the VFS Executive are described in this section. Syntax is detailed in Appendix A.

#### 3.1 BASIC FEATURES OF THE EXECUTIVE

The Executive is driven through a keyboard dialogue with the user. A series of commands may be grouped into an "indirect command file" and later called up for execution; these files may be chained together.

Comments may be inserted into command files; when encountered, they are simply echoed on the terminal. This is useful for displaying status messages or user prompts. A facility is also provided whereby execution of the command file may pause pending a continuation signal from the user; this allows, for example, preparation of a particular video tape sequence for loading into the VFS memories. (The VFS input ports accept video input which is converted to digital information. Since less time is required to

---

\*UNIX is a registered trademark of Bell Laboratories

produce a video tape than to master a video disc, we have used a 3/4" video cassette player as an input device. Although this method lacks computer controllability and random frame access, it has permitted the testing of VFS functions while the video disc is in production.)

UNIX commands may be entered without terminating the Executive, by means of a special escape character.

An interrupt key may be depressed at any time. This causes the current operation to stop, and the user is returned to the top command entry level of the Executive. (This interrupt feature has no effect on the VFS microsequencer. A VFS microprogram must always be running, or the frame store memories will not be refreshed and will begin to decay. Usually, a display program will be running, which simply reads pixels from frame store memory and sends them to the digital-to-analog convertors for display on a monitor.)

On-line help is available for all of the Executive commands, for special features (command files, UNIX commands, etc.), and for proper syntax in specifying VFS memories as command parameters.

### 3.2 VFS EXECUTION COMMANDS

These three commands affect the execution of the VFS microprograms.

#### 3.2.1 RUN Command

The RUN command starts the VFS microprogram controller. A start address may be specified, or the currently defined start address, maintained in a variable by the Executive, is used.

#### 3.2.2 HALT Command

The VFS microprogram controller is halted. (This will cause the frame store memories to decay.)

### 3.2.3 RESET Command

The VFS microprogram controller must be synchronized with the raster scan of the display monitor, or the picture will drift out of proper justification. Synch is occasionally lost when accessing certain memory locations, and may be restored by the RESET command.

## 3.3 MEMORY ALTERATION

Each of these commands prompts for a series of memory specifications. Memory ranges are specified by giving a bounding rectangle or, for DRAW and ERASE, a pair of endpoints. For frame store memories, a bitplane mask is also specified. The bitplane mask is an 8-bit number which identifies the subset of the color group's 8 bit planes which will be affected by the command.

### 3.3.1 SET Command

Sets all bits in the specified memory range to 1's.

### 3.3.2 CLEAR Command

Clears all bits in the specified memory range to 0's.

### 3.3.3 DRAW Command

This command is used for frame store memories only. It accepts a pair of endpoints and draws a line between them by setting all bits in the specified bitplanes to 1's.

### 3.3.4 ERASE Command

The complement to the DRAW command; the specified range of the appropriate bitplanes is cleared to 0's.

## 3.4 DUMPING DATA

Memory contents may be viewed on the screen, sent to the printer, or stored in a digital file for later reloading.

#### 3.4.1 DUMP Command

The DUMP command prompts for memory specifications which are the same format as those used by SET and CLEAR. Any combination of memory data may be dumped to one single file. If a range of microprogram memory is dumped, a start address may be stored in the dump file as well.

#### 3.4.2 LOAD Command

A previously dumped file is loaded into memory. There is an option to clear all memories before the file is loaded. Also, a microprogram may be loaded and immediately run if a start address was stored in the file.

A microcode assembler has been developed to aid in the development of microcode programs. The assembler produces object modules in the same format as files created with the DUMP command. If an assembler object module is loaded, a relocation address may be specified. Assembled microprograms have default origins, but are dumped by the assembler with relocation information so that any arbitrary address may be specified at load time.

### 3.5 THE VFS INTERACTIVE DEBUGGER

The VFS Interactive Debugger (VID) is entered through the Executive. It contains its own input processor because it operates in raw input mode, meaning that a number of keys in addition to the Return key can act as command breaks.

The primary purpose of VID is to obtain immediate feedback of individual byte changes in the various VFS memories. Microprograms may be altered while they are running, which is a great aid in debugging. It is also possible to make slight changes in coefficient memory while the coefficients are being used for video processing--this aids in debugging zoom algorithms.

VID maintains a screen display of certain key memory locations--display and write masks, A/D settings, etc. Most of these locations may be written to, but the VFS architecture does not permit their values to be read back. Therefore, a buffer is reserved to hold the latest value written into each such location.



There is a special input mode which allows the user to move the cursor among the "register windows" displayed on the screen, changing any or all registers. The new values are written to the VFS when the user exits this special mode.

Two-character mnemonics are provided for many important control registers, so that the user does not have to type in the full 6-digit address for these frequently used locations.

VID provides the lowest level of interaction with the VFS--individual bytes may be altered. However, certain key functions are clustered for convenience and are provided by a single command. For example, the screen translation offset consists of a 9-bit pixel offset and an 8-bit line offset. Four additional bits specify whether to blank the screen wraparound, and if so, which portions to blank. Thus, a total of 21 bits are necessary, but the size of a VFS byte is only 8 bits. To spare the user the task of properly breaking down the 21 bits into three 8-bit registers, a command is provided which accepts a line and pixel offset pair and sets the three registers accordingly.

Since the video disc player is used as the primary source device for images loaded into the VFS, a command is provided under VID for searching the disc for a particular frame number.

The syntax for indirect command files and UNIX command execution is carried over from the Executive. A mechanism is also provided for command file pauses, although its syntax differs from the Executive due to the special effects of many characters which cause command breaks. The syntax is fully described in Appendix A.

## SECTION 4

### APPLICATION DATABASE STRUCTURE

The tangible elements of the application data base are the video disc (containing all of the actual maps) and a set of data files which describe the video disc layout, the set of features and their colors, and all relevant VFS settings for loading the maps. The physical structure of the database is the result of a particular conceptual organization imposed on the mapped region.

#### 4.1 CONCEPTUAL ORGANIZATION

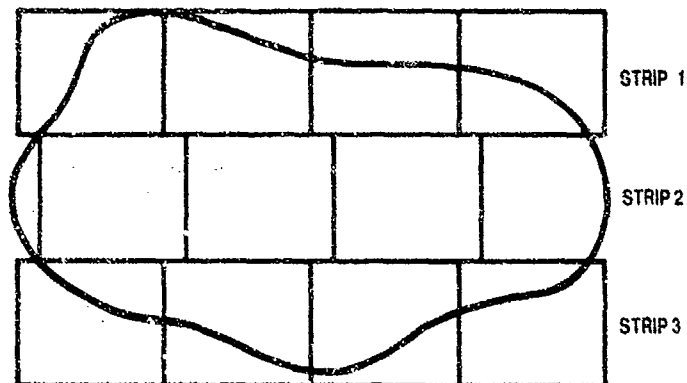
A central aspect of the application of the VFS as a terrain map display system is the use of the individual planes of memory to hold separate map features. These planes are then selectively combined at display time to produce custom-tailored map displays. The feature separation plates used in the preparation of paper maps are divided up into screen-sized pieces and photographed, and these photographs are stored on the video disc.

The fundamental division unit within the map database is the map level. This is the scale, or magnification, level of the map, determined by the paper map from which the separation plates were taken.

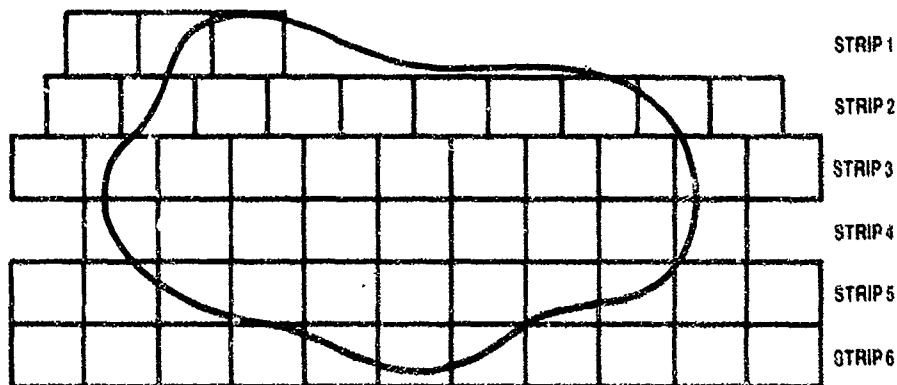
Each map level is divided up into equal sized pieces which will ultimately be loaded into the VFS. The level is first broken down into horizontal strips, each of which may start at a different longitude, depending on the boundaries of the region of interest. Each strip in a level has the same height (latitude).

The strips are then divided into photos of equal width. A photo is geographically the smallest piece of a map which is handled. Due to the feature separation, a photo is physically a sequence of separate frames on the video disc, each frame holding a separate feature or background image. Logically, however, it is convenient to conceptualize all of these frames for the same small geographic region as a single unit, a photo.

The terms "level," "strip," "photo," and "frame" will be used throughout this section and the next and should be clear to the reader at this point. Figure 2 is an illustration of a hypothetical two-level map region, showing strips and photos at each level.



LEVEL 1 -- 1:50,000



LEVEL 2 -- 1:25,000

1A-66,517

Figure 2

Strip and photo breakdown of a hypothetical region  
at scales 1:50,000 and 1:25,000

Although the actual map feature set differs from level to level, there must be some continuity between the levels. If a user decides to change the color of roads from black to red, it would be disorienting if they were to switch back to black when he changed to a different scale level. Therefore, feature colors must be maintained in a global list so that changes apply consistently at all levels.

Because feature colors are implemented through a color lookup table, it is possible to independently specify the color of feature intersections. For example, a user may wish to see rivers in blue, roads in black, and the intersection of the two in red, to indicate bridges. As with feature colors, the feature intersection color table is maintained globally, for consistent color mapping.

A mechanism for controlling feature precedence must also be included in the database. Thus, if rivers, in blue, have a higher precedence than contour lines, in brown, any pixels which include both rivers and contour lines will appear blue (unless a feature with higher precedence than rivers is present, or a special intersection color has been selected).

#### 4.2 VIDEO DISC LAYOUT

Any individual photo in a level will include some number of foreground frames and background frames. The foreground frames are the features which are displayed through the color lookup table, and can therefore be stored as black and white on the video disc. Three methods of creating the color background are possible with the VFS. Individual bitplanes of the red, green, or blue color memories (not the foreground bitplanes, which are color mapped) may be set according to a black and white image; this method involves a series of black and white frames on the video disc for the background. Or, a single full color frame may be read in to the color memories. Finally, three individual color frames may be present, each of which is read in to a different color memory. So, the three variables which describe the number of video disc frames for a particular photo are:

- f - number of foreground frames
- b - number of black and white background frames
- c - number of color background frames (1 or 3)

Note that both the black and white and the color options for background frames may be present on the disc; for any given application, however, one of these methods must be chosen over the other.

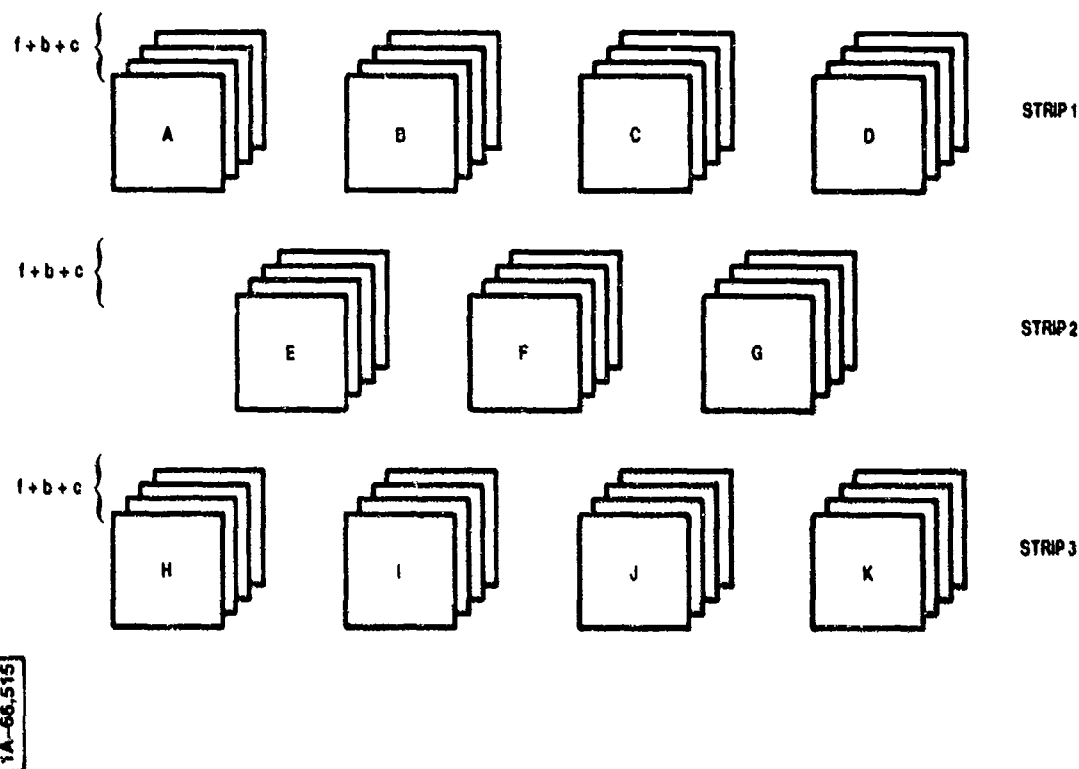


Figure 3

### Frame Expansion of Level 1 of the Region of Figure 2

The values of  $f$ ,  $b$ , and  $c$  are the same for all photos at a given level. Also, the order of the features on the video disc is the same. For example, frame 1 will be rivers, frame 2 roads, etc. for all photos on the same level. Figure 3 shows the photos of level 1 in Figure 2 expanded into individual frames.

For faster map loading, all frames for a particular photo should be grouped together on the disc. However, the disc mastering process requires that color frames be grouped together on the disc separate from black and white frames. So, the frames of a photo are split into two groups: one for the foreground and background black and white frames (f+b), and the other for the single or three color frames (c). Groups of frames from adjacent photos in a strip are placed next to each other on the disc. Figure 4 shows the disc layout of our sample map level.

Any feature of any photo may be located on the disc by knowing the values of f, b, and c, the strip height, the photo width, the latitude of the first strip, the longitude of the first photo in each strip, and the first frame number of each strip. The procedure is illustrated by attaching concrete values to our map region.

Suppose the relevant values are as follows:

$$f = 14 \quad b = 2 \quad c = 1$$

$$h \text{ (strip height)} = 4' 0''$$

$$w \text{ (photo width)} = 5' 30''$$

<u>strip</u>	<u>longitude</u>	<u>first foreground frame</u>	<u>color frame</u>
1	105° 0' 0" E	2400	17000
2	105° 3' 0" E	2464	17004
3	105° 1' 0" E	2512	17007

To find the photo containing the location 32° 54' N, 105° 15' E, and load the third foreground frame, we would first determine the strip number. This is the number n such that

$$1 + nh$$

is greater than the desired latitude, but

$$1 + (n-1)h$$

is less. (Some sign adjustment is necessary depending on position relative to the equator.) In our case, n is 2.

The next step is to determine the photo in the strip. The formula is similar to that for finding the strip, except the strip longitude replaces l and the photo width (w) replaces h. The photo number turns out to be 3.

Now, the frame number of the third foreground frame of the third photo in the second strip may be calculated:

$$\begin{aligned} & (\text{first strip frame}) + ((\text{photo } \#) - 1)(f+b) + (\text{frame } \#) - 1 \\ & = 2464 + (3 - 1)(14+2) + 3 - 1 \\ & = 2498 \end{aligned}$$

1A-66,513

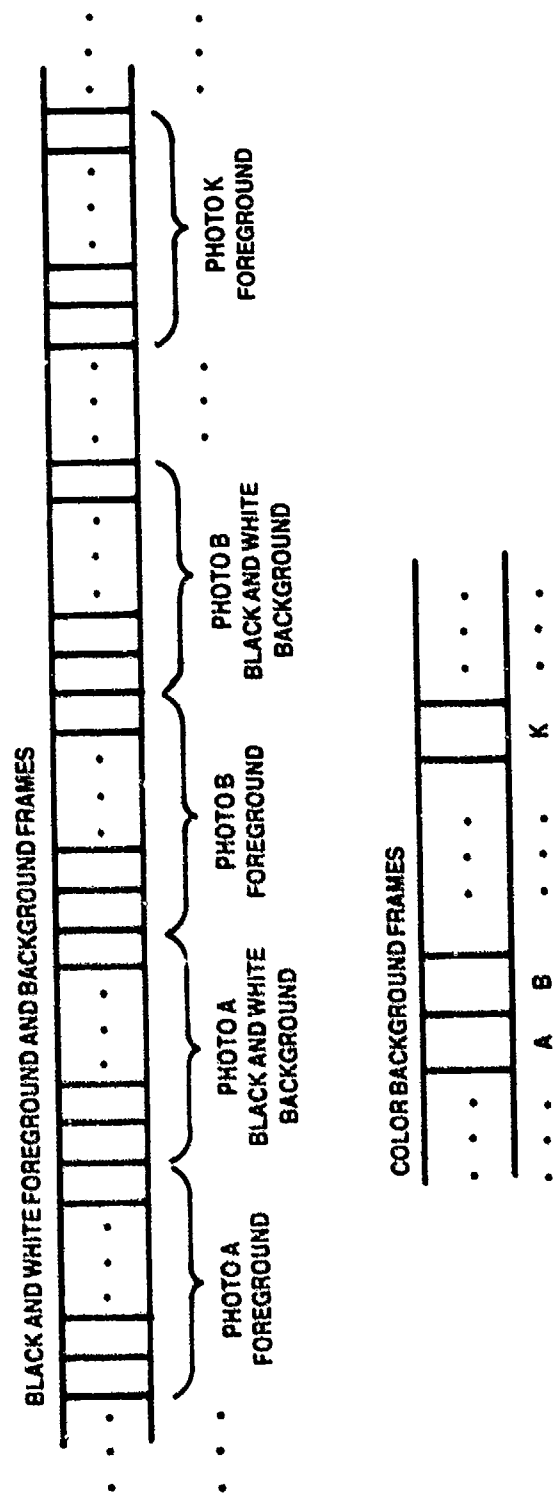


Figure 4. VIDEO DISCLAYOUT OF THE PHOTOS IN FIGURE 3.  
(EXAMPLE SHOWS ONE COLOR BACKGROUND FRAME PER PHOTO.)

### 4.3 DATA FILES

The parameter values used in the calculations of the last section must all be stored in a data file specifically relating to the video disc. Likewise, lists of features and frame numbers must also be maintained for each level. (A user is not apt to ask for "foreground frame 3," but to ask for, e.g., "rivers"; the data base would then be consulted to determine that at the user's current level, rivers are in the third frame.)

Three separate files are maintained for the three types of bookkeeping which must be performed: a video disc directory, a feature color table, and an application definition file.

#### 4.3.1 The Video Disc Directory

The video disc directory contains all of the information needed to locate individual map photos on the corresponding video disc. As illustrated in Figure 5, it is broken down by level, and by strip within each level. A file "entry" contains all of the information which remains constant for a given level. For each strip in each level, there is a "sub-entry" containing particular information for the strip.

The specific information contained in an entry is listed in Figure 6. The nature of this data is described in the paragraphs which follow.

**Scale value.** The reduction scale factor when the map is displayed on the color monitor. This is not necessarily the same scale as the paper map from which this level of photographs was made, because the field guide used for the photographs is not always the same size as the display screen. (By using a field guide half the size of the screen to photograph a 1:500,000 scale paper map, the individual photos, when seen on the screen, are at a scale of 1:250,000.)

**Scale range.** The lower and upper scale limits for which this map level will be used. If a zoom factor is specified which falls within these values, a software zoom is performed in the VFS. If, however, the limits are exceeded, the application controller will switch to a different map level.

**Number of foreground frames per photo.** The number of black and white "feature" frames of each photo in each strip of this level.



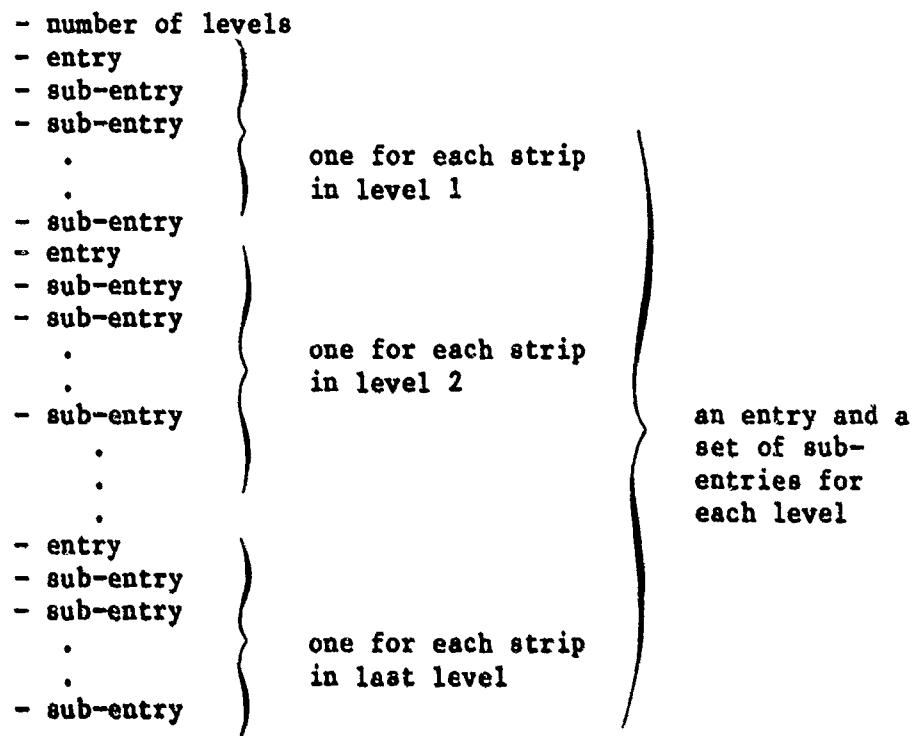


Figure 5

### Video Disc Directory Format

scale value  
 scale range  
 number of foreground frames per photo  
 number of black and white background frames  
     per photo  
 number of color background frames per photo  
 photo height  
 photo width  
 latitude of first strip  
 number of strips

Figure 6

### Data Elements of a Video Disc Directory "Entry"

Number of black and white background frames per photo. The number of black and white frames, if any, for the color background of each photo in each strip of this level.

Number of color background frames per photo. This value applies to each photo in each strip, and will be 1 for full color background frames and 3 for red/green/blue separations.

Photo height. The height in degrees, minutes, and seconds of each photo. This is the vertical distance between any two adjacent strips.

Photo width. The width in degrees, minutes, and seconds of each photo. This is the horizontal distance between any two adjacent photos in a given strip.

Latitude of first strip. The vertical position of the first strip, in degrees, minutes, and seconds north or south. The position of each successive strip can be calculated by adding the photo height. An arbitrary standard point on the strip may be used for positioning--upper left corner, center of left edge, center of first photo, etc.--but it must remain standard throughout the application.

Number of strips. The number of strips of photos at this level. For each strip, there is a sub-entry with detailed data.

longitude of first photo  
number of photos  
first foreground frame number  
first color background frame number

Figure 7

Data Elements of Video Disc Directory "Sub-entry"

The contents of a sub-entry are listed in Figure 7; these are explained in the paragraphs which follow.

Longitude of first photo. The horizontal position of the start of a strip, in degrees, minutes, and seconds east or west. Upper left corner, photo center, etc. may be used as a reference, but must remain standard.

Number of photos. The number of photos in the strip. With the photo width, this indicates exactly how far the strip extends.

First foreground frame number. The actual video disc frame number of the first foreground frame of the first photo in this strip. All frames for the strip are in consecutive order on the disc. Black and white background frames follow each photo's foreground frames.

First color background frame number. The starting location of the color section of the disc for this strip. Each photo in the strip may have one or three color frames, as indicated by the value in the main entry for this level.

#### 4.3.2 The Feature Color Table

The feature color table contains color values and precedence levels of all features defined in the application, as well as all feature intersection lists. The format of the feature color table is shown in Figure 8.

number of features	
feature name	
feature color	} repeated for each feature
feature precedence	
frame map	
number of intersections	
intersection color	} repeated for each intersection
strong/weak	
intersection precedence	
feature mask	

Figure 8

Feature Color Table Contents

**Feature name.** An alphanumeric identifier for the feature. Some applications may allow feature specification by name (in fact, it is expected that most will); the names appear here in a permanent order.

**Feature color.** A three digit code, each digit ranging between 0 and 7. These digits specify the blue, green, and red intensity values (BGR, left to right), and are filled in to color mapping memory.

**Feature precedence.** A unique number specifying the feature's display precedence; the lower the number, the higher the precedence. Features fill color mapping memory in reverse precedence order, so that the feature with the highest precedence (lowest number) is filled last and color mapping memory locations previously set are reset with the color of this last feature.

**Frame map.** An inverted list indicating the feature's foreground frame number at each level, relative to the other features. (This is not an absolute video disc frame number.) If a feature is not present at a particular level, a special value is put in the map for that level.

**Intersection color.** The color (BGR) of the particular intersection specification. Intersections are filled into color mapping memory after all of the features have been filled, so that intersection colors supercede feature colors.

**Strong/weak.** A flag indicating whether the intersection is strong or weak. Setting up a strong rivers/roads intersection will cause that intersection color to appear for rivers/roads/cities, rivers/roads/vegetation, etc. Weak intersections, on the other hand, cause pixels to be affected only if the exact list of features is present.

**Intersection precedence.** The precedence of the intersection specification, relative to the other intersections.

**Feature mask.** A bit mask indicating which features comprise the intersection. If feature *i* is included, bit *i* is set in the mask. (A feature's number is simply its relative position in the feature table, because the order of the table never changes.)

### 4.3.3 The Application Definition File

The application definition file contains all of the information concerning actual settings of the VFS for loading video disc frames. All of these settings are map level specific. The format of the application definition file is shown in Figure 9.

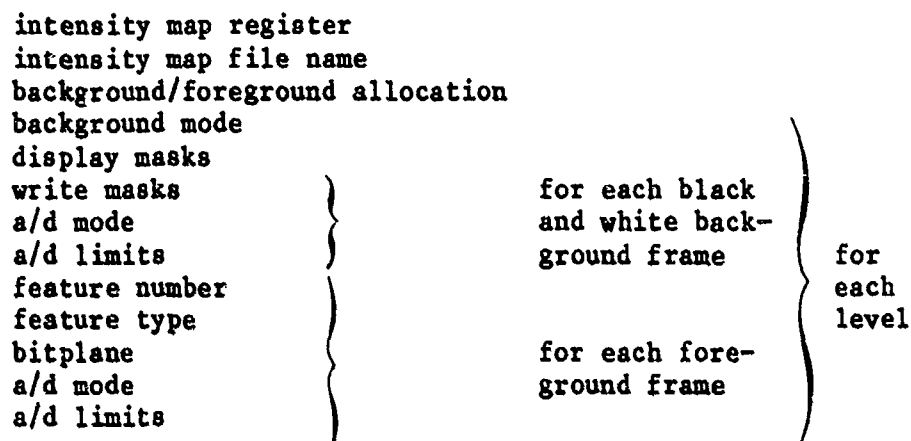


Figure 9

#### Application Definition File Contents

**Intensity map register.** This is the actual value of VFS memory location 166000 (octal), which indicates the intensity map mode. The eight bits of the register are divided among the three colors as follows:



where, for each color, the pair of bits indicates a quadrant of intensity mapping memory. Currently, these quadrants are loaded with values to perform the following intensity corrections:

- 00 = gamma correction
- 01 = inverse gamma
- 10 = uncorrected (linear)
- 11 = inverse linear

**Intensity map file name.** The name of a VFS dump-format file which will be loaded into intensity mapping memory.

**Background/foreground allocation.** The contents of VFS memory location 177773 (octal). This register indicates how the VFS memory planes are to be split up between background and foreground. The interpretations are as follows:

- 0 = 12 background / 12 foreground
- 1 = 15 background / 9 foreground
- 2 = 18 background / 6 foreground

So, with a value of 2, planes 1 and 2 of each color group are available for foreground features which are mapped through the color lookup table; with a value of 0, planes 1 through 4 of each color are used for features, and the total remaining 12 frames make up the full-color background image.

**Background mode.** This flag indicates whether the color or the black and white frames will be used to form the background of photos at this level.

**Display masks.** A trio of 8-bit masks indicating which background planes will be displayed. The masks are placed in the three registers starting at octal location 177770. These are useful for fine adjustments in the color of the background. The maximum values vary depending on the background/foreground allocation.

**Write masks.** A trio of 8-bit masks provided for each black and white background frame indicating which memory planes will be loaded with the image in the frame. They are placed in the three registers starting at octal location 177760.

**A/D mode.** A four-bit code indicating the A/D convertor settings, provided for each black and white background frame as well as each foreground frame. The value is placed in the A/D mode register, octal location 177740. The bits have the following interpretation:

**Bit 0 - HARDLIMIT**

Indicates that the A/D board should pass video data within the limits set in the limit registers; digitized video intensity data with numeric values outside the values in the limit registers are filtered out.

**Bit 1 - SATURATE**

Indicates whether to pad the screen with full video on (white) or full video off (black) when intensity levels are outside limits.

**Bit 2 - ANDOR**

When set, video data is passed only if all three channels are within the limits specified by the limit registers.

**Bit 3 - SMEAR**

When the ANDOR bit is not set, the SMEAR bit determines whether to output video on all three channels (SMEAR = 1) or only those channels within limits (SMEAR = 0). Of ANDOR is set, the state of SMEAR is irrelevant since all the input must be within the limits to generate an output.

**A/D limits.** These values are loaded into the six registers beginning at octal location 177741 and represent, in order, the red lower limit, the red upper limit, the green lower limit, the green upper limit, the blue lower limit, and the blue upper limit. Each value is 6 bits. Limits are provided for black and white background frames, and for foreground frames.

**Feature number.** For each foreground frame, there is a feature number which is an index into the feature color table.

**Feature type.** Up to a maximum of 12 features (and possibly only 9 or 6, depending on the background/foreground allocation) may be loaded into the frame store memories. The feature type indicates whether a feature is loaded and displayed (type = MANDATORY), loaded but not displayed (type = PRIMARY), or not loaded at all (type = SECONDARY). The feature types will generally change frequently while an application is being run, as different features are selected for display. A SECONDARY feature may only be selected if there is an available bitplane into which the feature may be loaded. It may be possible to remove a PRIMARY feature (i.e., change its type to SECONDARY) to make room for the new feature. If, however, all foreground bitplanes contain MANDATORY features, the user must erase one of the features to make room for the new one (or the application software may include some automatic method of choosing a feature to be removed).

**Bitplane.** For each foreground frame, the bitplane into which the frame is loaded must be indicated. If the feature is SECONDARY, of course, there is no bitplane number. The frame store memory planes are physically divided into three color groups, and are numbered from 1 to 8 in each group. The first 2, 3, or 4 planes in each group are used as color-mapped foreground frames, depending on the background/foreground allocation. The remaining high-order

bitplanes of each color group contribute to the digitized full color background image. The numbering of the bitplanes, for each possible background/foreground allocation, is illustrated below:

Background/Foreground  
Allocation

Bitplane "Map"

	8	7	6	5	4	3	2	1	
18/6	-(background planes)-						4	1	Red
							5	2	Green
							6	3	Blue

	8	7	6	5	4	3	2	1	
15/9	[ (background planes) ]					7	4	1	Red
						8	5	2	Green
						9	6	3	Blue

	8	7	6	5	4	3	2	1	
12/12	-- (back- --	10	7	4	1	Red			
	-- ground --	11	8	5	2	Green			
	-- planes) --	12	9	6	3	Blue			



## SECTION 5

### APPLICATION DEFINITION

#### 5.1 OVERVIEW

The database described in Section 4 is defined through an interactive process by the application programmer. The name of the data entry program is "define", and it is invoked with a unique application name as an argument. Once the application programmer has keyed in the database description, the program produces an ASCII text file whose name consists of the application name with the suffix ".ed". This file is an intermediate, editable file which simply consists of all of the programmer's entries, with a brief description of each data element. This allows minor database changes without going through the entire entry cycle.

The actual database is produced by invoking "define" again, with the application name as a first argument, and any arbitrary second argument. The --.ed file will be read and the three files which comprise the application database will be produced; the name of each will consist of the application name and an appropriate suffix: ".vdd" for the video disc directory, ".fct" for the feature color table, and ".adf" for the application definition file.

#### 5.2 DATA ENTRY

The data entry cycle is controlled by a screen formatting utility which makes use of screen attributes such as half-bright, inverse video, and inverse half-bright to enhance the interaction. Typically, an input "panel" appears before the user for a related set of database specifications. All text and prompts appear half-bright. Variable prompts--such as level number, strip number, etc.--are in full bright mode. All user input areas on the screen appear at this time in inverse half-bright mode. As the user progresses through the entries on the screen, his current input area is displayed in inverse video full bright mode. Once each entry has been completed, that input area is locked and switched to normal video (non-inverse, full bright). The full brightness of these filled in input areas thus stands out in contrast to the half-bright prompts.

In some cases, specific prompts or messages are added to the screen during the input cycle. These may be error messages, specific data selection criteria instructions, etc. If a particular

screen panel is to be used repeatedly for multiple levels, features, etc., all such text is removed, and the input areas are cleared and reset to inverse half-bright for the next set of data.

In a few cases, more than one input area will be activated at a time for entering related elements of data. In such cases, the programmer can move back and forth among the fields by pressing specifically designated "function keys" on the keyboard.

The paragraphs which follow briefly trace the steps involved in the entry cycle. Details may be found in the Application Definition User Manual, in Appendix B of this report. Screen layouts are also included in the manual.

#### 5.2.1 Number of Levels

The programmer is initially prompted for the number of map scale levels in the application. This is a simple interaction with no elaborate screen formatting.

#### 5.2.2 Video Disc Directory "Entry"

Approximately two-thirds of the screen is formatted for entering the data which makes up a video disc directory "entry", as defined in paragraph 4.3.1 above. Since such an "entry" exists for each level, a field on the screen displays the level number.

#### 5.2.3 Video Disc Directory "Sub-entry"

After the information for an "entry" has been provided, the top portion of the screen is locked and the bottom portion is formatted for accepting the "sub-entry" information, as defined in paragraph 4.3.1. This screen area is repeatedly reset for each "sub-entry" (each strip of the current level) until all strips have been specified. At this time, the bottom third of the screen is cleared, and the top portion is reset for the next level "entry".

#### 5.2.4 Number of Features

This information is entered through a simple prompt-response exchange (i.e., with no elaborate screen formatting), to determine the number of features for the global feature color table.

#### 5.2.5 Features and Colors

For each feature, the programmer enters a unique feature name (up to 14 characters) and a feature color (a blue/green/red octal triplet). The order in which the features are entered determines the precedence, with the first feature specified having the highest precedence and the last having the lowest. The "frame map" defined in paragraph 4.3.2 is not entered at this time, but is computed from information entered later.

#### 5.2.6 Feature Intersections

Once the indicated number of features have been specified, a screen panel appears for intersection specifications. The programmer enters a color, indicates whether the intersection is strong or weak (as defined in paragraph 4.3.2), and types a list of feature names. The screen is then reset and a new intersection may be specified. Intersection entry is terminated by pressing the "return" key without entering data; if the programmer does not wish to specify any intersections, he can do this when the screen panel first appears.

#### 5.2.7 Application Definition File Constants

After intersection specification, the screen is formatted for those elements of the application definition file which are constant for an entire map level (e.g. intensity map register, background mode, etc.).

#### 5.2.8 Black and White Background Frames

If there were any black and white background frames specified in the video disc directory for the current level, the screen is formatted so that the write masks, a/d mode, and a/d limit register values may be entered for each frame. If there were none, this screen panel is skipped.

#### 5.2.9 Foreground Frames

Finally, the screen is formatted to accept the information for each foreground frame. A feature name is entered for each foreground frame, and the "frame map" in the feature color table is

properly annotated. After all foreground frames have been specified, the screen clears and is reformatted for the next level's application definition file constants.

### 5.3 THE DATABASE REPORT

Once an application database has been defined, the application programmer may print a database report. The program "report" is invoked with an application name as an argument. It prints the information contained in the three data files to the standard UNIX output file, for display on the screen or redirection to the printer. The three files are separated by form feeds, as are the individual levels, to produce separate pages if output is directed to the printer. A title page is also provided.

Feature precedence is displayed with each feature. In general, these values will just appear as a list of consecutive numbers, because features are originally entered into the file in precedence order. While an application is being run, the precedence of the features can be changed. It is conceivable that a command may be added to the application controller which would allow the database to be re-written from the internal values being used, thereby scrambling the feature precedence.

## SECTION 6

### INTERACTIVE APPLICATION CONTROL

The purpose of the interactive application control package is to provide a facility for presenting clean, effective demonstrations of the VFS terrain map display system. Command functions are clustered at a "higher" level than the VFS Executive program. The user has no access to VFS registers, no direct bitplane control, and no microprogram debugging capability. A few relics do remain--for example, the ability to restore sync--but these exist only because the users of this demo package are VFS project members. A pure application control package, such as the one described in the next section, would be free of any reference to the VFS architecture.

The commands of the application control package are described here only briefly; detailed syntax may be found in Appendix C, the Application Controller User's Manual.

#### 6.1 INVOCATION FORMS

The application control program is called "application" and may be invoked in three different ways:

- 1) application <application-name>  
will activate the named application by loading the database, and will read the first map into memory and display the appropriate features on the screen.
- 2) application <application-name> -  
will load the database, but will not disturb the VFS memories. The video disc will never be accessed for map images. This mode is for images which have just been loaded from video tape or from magnetic disk under control of the VFS Executive.
- 3) application <application-name> <image-file-name>  
will load the database, and then load the indicated image file into VFS memory. The image file is a digital file dumped by the VFS Executive.

## 6.2 FEATURE DISPLAY CONTROL

Once the application has been activated and its database loaded, features may be selected or removed from the screen by name.

### 6.2.1 ADD Command

The ADD command displays a feature if it was not previously displayed. (That is, the feature type is changed to MANDATORY.) Feature color and precedence are in no way affected by the ADD command.

If the selected feature is a SECONDARY feature (i.e., not resident in memory), it must be loaded from video disc. This, of course, is only done if invocation form 1 was used. If all bitplanes are filled by other MANDATORY and PRIMARY features, a message is printed on the CRT screen.

### 6.2.2 ERASE Command

The ERASE command disables the display of a selected feature (its type is changed from MANDATORY to PRIMARY). The feature remains loaded in memory so that it may be instantaneously displayed at some later time.

### 6.2.3 REMOVE Command

This command frees the bitplane occupied by the specified feature by changing the feature's type from MANDATORY or PRIMARY to SECONDARY. A new feature may then be loaded from the video disc using the ADD command if all bitplanes were previously full.

## 6.3 COLOR AND PRECEDENCE CONTROL

### 6.3.1 COLOR Command

The color of a feature may be changed simply by specifying the feature name and new color (as a BGR triplet). The feature does not have to be displayed at the time.

### 6.3.2 COLOR Command for Intersections

Intersection colors are established or changed by giving the color, a strong or weak indication, and a list of features.

### 6.3.3 PRECEDE Command

Feature precedence is altered by specifying two features; the relative precedence of all features is then adjusted so that the first feature has precedence just above the second. One feature alone may be specified, in which case it receives highest precedence.

## 6.4 PICTURE MANIPULATION

### 6.4.1 ZOOM Command

Various integral and fractional zoom factors are supported; the microcode is resident in the VFS at all times so that a change in the scale factor is simply a matter of changing the VFS start address. If an unsupported magnification factor is chosen, a message is printed on the CRT screen. Currently implemented factors are: 1 (i.e., no magnification), 1.25, 1.5, 2, and 4. Other factors will be added but have not yet been microcoded.

### 6.4.2 TRANSLATE Command

The display is instantaneously shifted to the indicated line and pixel offset.

### 6.4.3 SLIDE Command

The display is shifted a relative number of lines and pixels, in a smooth sliding motion.

### 6.4.4 JOYSTICK Command

The user translates the screen in n-pixel and n-line increments (n is a command argument whose default value is 1) by pressing the keys on the numeric keypad, thus simulating the effect of a joystick device. Direction of translation is determined by the key's position relative to the '5' key in the center of the pad. Pressing '0' terminates the JOYSTICK command.

## 6.5 VIDEO DISC INTERFACE

Two commands are provided which permit the display to be switched between the stored map and the analog video disc image, and allow the video disc to be scanned for a particular frame.

### 6.5.1 DISC Command

When used without an argument or with the word "ON", the display is switched to the video disc. When used with "OFF", the map display is restored.

### 6.5.2 SEARCH Command

Any frame on the video disc may be located by entering the frame number with this command.

## 6.6 MISCELLANEOUS

The application controller supports UNIX command execution (preceded by '!'), indirect command file input (preceded by '<'), and comments (preceded by ';').

### 6.6.1 QUERY Command

This command lists, in precedence order, each feature name, its color, and whether it is displayed, loaded but not displayed, or not loaded at all. Intersections are then listed in precedence order, with strong/weak indication and intersection color. If the image is translated, the offset is displayed, and if it is zoomed, the zoom factor is indicated.



#### 6.6.2 LOAD Command

The LOAD command, a VFS Executive holdover, loads the indicated VFS dump format file. Its main use in the demo environment is for loading different coefficients to alter the zoom algorithm (e.g. repeat pixel vs. bilinear).

#### 6.6.3 SYNC Command

This command restores synchronization between the microsequencer and the raster scan; its function is identical to the VFS Executive RESET command, discussed in paragraph 3.2.3.

#### 6.6.4 PAUSE Command

The behavior of the PAUSE command is the same as for its namesake in the VFS Executive. It provides the ability to run automatic demonstration sequences from indirect command files.

## SECTION 7

### INDIRECT APPLICATION CONTROL

As discussed in the introduction, the ultimate configuration of the VFS will be a dual processor environment where one CPU (possibly a microcomputer) always serves as the VFS host, and a second CPU handles the intelligence application. The VFS contains its own microprocessor for graphic and image processing functions, and uses few host CPU resources. The only requirement of the VFS host is a set of routines which manipulate a background map through appropriate control of the VFS registers, and which also control the video disc player. If a single host is used, these routines are invoked by a co-resident human-machine interface package. In a dual processor configuration, the same routines may be used, but are invoked via messages passed from the application processor. (Refer again to Figure 1.)

For complete generality, the application processor should be spared as much "bookkeeping" as possible. All tables, feature lists, etc.--the so called application database--should be maintained on the VFS host processor. Thus, the software may be derived by adapting the interactive application controller to a dual processor environment.

For speed, however, a message protocol system must be developed which involves the transfer of as few bytes as possible for the invocation of any given routine. Such a consideration disallows the transmission of feature names as arguments to commands; feature index numbers can be transferred much more quickly. Likewise, upon completion of a particular command, the VFS host processor should pass a return code to the application processor. Any error number returned to the application software package will have to be interpreted and acted upon to determine what further operator actions are necessary. Passing a full error message would be too time consuming.

The CPU running the intelligence application, then, must "know" about all possible error conditions, as well as the map's feature names, the current scale level, coordinate position, etc. Such a requirement is not overly restrictive, although it does violate the modularity objectives established for an "ideal" dual processor configuration. The application package will be interfacing with the user, in whatever way is best suited to the application--keyboard, touch-screen, pointing devices, voice, etc.--and will therefore have to maintain a list of the map features available. In addition, it

is likely that the package will have knowledge of the map level and geographical coordinates of the currently displayed background map, in order to coordinate the display of overlaid intelligence information.

At the present time, the interprocessor message handling package has not yet been implemented. The major goal of the project thus far has been the refinement of a high level, interactive package for demonstrating, exercising, and further debugging the VFS in a terrain map display environment. Future plans include the design and implementation of a simple (hypothetical) intelligence display application on another processor, and full development of the interprocessor link. This work will not be started until the initial test video disc (described in the next section) has been produced.

## SECTION 8

### INITIAL DEMONSTRATION VIDEO DISC

An optical video disc is now in production which will make it possible to test all of the features of the VFS terrain map display system. Until now, we have used a 3/4 inch video tape containing one set of map separations, and this has permitted us to test all VFS features except switching between adjacent map photos and between map levels. A full scale database on video disc will provide us with the random access search capabilities to demonstrate these functions.

#### 8.1 GEOGRAPHIC AREA COVERED

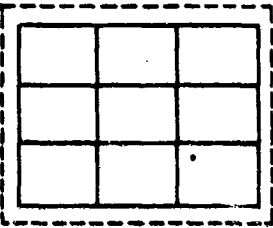
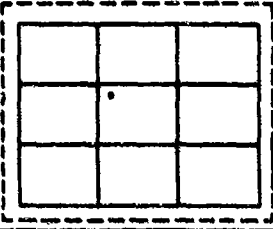
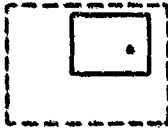
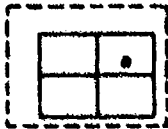
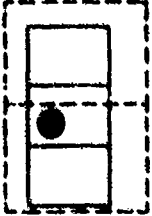
The maps used for the video disc are Defense Mapping Agency (DMA) paper maps of the area around Fulda, West Germany. The maps are at four different scales: 1:1,000,000, 1:500,000, 1:250,000, and 1:50,000. At the 1:50,000 scale, there are two DMA maps which are vertically adjacent. The 1:250,000 scale map will be photographed a second time at a 3:2 ratio to produce an effective 1:167,000 scale map. The relative geography and coverage of the maps is outlined in Table 1.

For levels 1, 2, and 5, the maps will be photographed with a 17" diagonal mask. Assuming a 17" diagonal CRT display area, this results in a preservation of the paper map scales. For level 4, an 11" diagonal mask will be used; the resulting magnification when the map is displayed on a 17" CRT yields an effective scale of 1:167,000.

Notice that for level 5, since the two paper maps are registered perfectly adjacent, they will be joined together and photographed in three parts.

---

\* Although we have been using video tape, the video disc interface has been working for quite some time. It is actually almost identical to the video tape interface, with an additional control line to transmit the frame search command.

LEVEL	SOURCE PAPER MAP	FIELD SIZE (DIAGONAL)	NAUTICAL MILES DISPLAYED ON A 17" SCREEN	EFFECTIVE SCALE	PARTITIONING, SHOWING APPROXIMATE LOCATION OF FULDA
1	ONCE-2 1:1,000,000	17"	180	1:1,000,000	
2	TPC-E2C 1:500,000	17"	90	1:500,000	
3	JOO NM32-5 FRANKFURT AM MAIN 1:250,000	17"	45	1:250,000	
4	JOO NM32-5 FRANKFURT AM MAIN 1:250,000	11"	30	1:147,000	
5	LS324 HÜNFELD LS524 FULDA 1:50,000	17"	9	1:50,000	

15-48-511

TABLE 1. COVERAGE OF GEOGRAPHIC REGION BY VIDEO DISC MAPS

1A-88,516

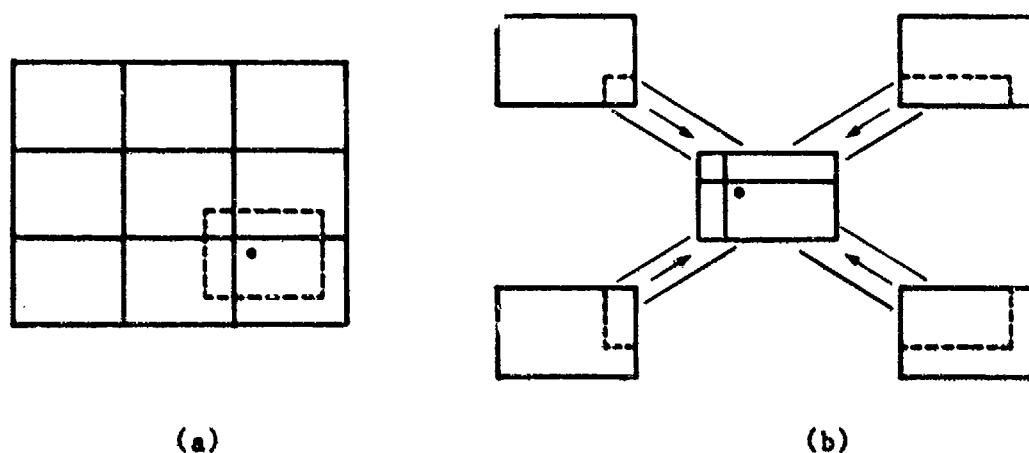


Figure 10

#### Creation of a Pseudo-map from Adjacent Areas

- (a) example of an area with 9 partitions, showing the desired pseudo-map
- (b) construction of the map by four individual partial writes to memory

It should be noted that although the region at a particular level is divided into screen-sized pieces, any point in the area can be displayed at the center of the screen. This is accomplished by loading adjacent maps into distinct areas of memory. The procedure is illustrated in Figure 10.

#### 8.2 MAP FEATURES

A perfect multi-level map system would have total consistency of features across all levels. As a user moves from one level of magnification to another, all of the features he was viewing should remain.

Unfortunately, the resources available for creating the database do not allow this. The ONC and TPC series DMA maps have separation plates available for the individual features. But the JOG and the L-series maps have color separation plates. In order to get individual features, copy negatives must be altered by hand--an expensive, time consuming process for maps with much detail. Where feasible, however, these manual separations will be made.

On the other side of the coin, certain related features may appear on the same separation plate at one level, and on different plates at another. If it is conceptually valid to deal with the features together, they can be combined at the level where they are separated by double-exposing the film during the photographic process.

Table 2 lists all of the video disc frames which are a part of the database. Instances of double exposure and manual feature separation are noted. More than one color background frame will be produced at some levels, for a wider range of options and applications. A total of 366 frames will be produced.

Feature no	Level 1	Level 2	Level 3	Level 4	Level 5
Color Back-ground 1	level area	250' -> 500' 2000' -> max			
Color Back-ground 2	level area shaded relief	250' -> 500' 2000' -> max shaded relief	vegetation shaded relief	vegetation shaded relief	
1	shaded relief	shaded relief	shaded relief	shaded relief	
2	open water / inland water (a)	inland water			
3	drainage	drainage	drainage (b)	drainage (b)	drainage (b)
4	vegetation	vegetation	vegetation	vegetation	vegetation
5	contours	contours			
6			roads, contours	roads, contours	roads, contours
7	roads	roads	roads (b)	roads (b)	roads (b)
8	culture	culture	culture	culture	culture
9	boundaries	boundaries			
10	cities	cities			
11	projection	projection			
12	grid	grid	grid (b)	grid (b)	
13	AD12	AD12			
14	zero - maximum elevation / isogonals (a)	zero - maximum elevation, isogonals			
15			zero, powerlines	zero, powerlines	
16	powerlines	powerlines			
17	special use	air info / special use (a)			
Total Frames	2 + 15 = 17	2 + 15 = 17	1 + 8 = 9	1 + 8 = 9	0 + 5 = 5
No. photos (see Table 1)	9	9	1	4	3
Total frames on video disc	153	153	9	36	15

Table 2 - Video Disc Frames

(a) - frame constructed by double exposing photographs of two different separation plates  
(b) - frame constructed by blacking out extraneous information from a more detailed separation plate



## SECTION 9

### SUMMARY AND FUTURE DIRECTIONS

#### 9.1 SUMMARY OF ACCOMPLISHMENTS

The software described in this report was developed over a period of approximately one year. The major immediate result of this effort has been the ability to present smoother, more effective demos which can be set up in a fraction of the time previously required. By creating an indirect command file under the application controller package, a series of map manipulations -- separated by a single touch of the return key -- can be integrated with a "viewgraph" style briefing. Previously, the Executive was used for all VFS control; feature selection was done by altering the bitplane mask registers, colors were changed by modifying individual locations of color mapping memory, and zooming was done by appropriately setting the start address register. Even without an indirect command file, these operations are greatly improved in the new interface, for example:

```
erase roads, railroads
add cities
color cities 477
zoom 1.25
```

In parallel with the software development, a procedure for properly photographing sections of paper maps was developed and tested sufficiently to prove the concept that map feature separations can be transferred to video disc in proper registration. As an intermediate result (the actual video disc being the ultimate goal), we now have a video tape of photographs of an actual map.

#### 9.2 FUTURE WORK

Using a video tape as input corresponds to a single-level, single-strip, single-photo map database, since only one map is available at any time. Once a full map database has been established on a video disc, the operations which could not be executed with the video tape will have to be implemented. The notions of translation and zooming will have to be expanded to encompass an area larger than a single map and a database "deeper" than one magnification level.

VFS microcode will be developed to piece together the proper sections of four adjacent maps, in order to bring any point to the center of the screen. This image-area translation will be combined with intelligent map photo selection to approximate continuous translation (real-time roam). Ways of achieving actual real-time roam are discussed in the next sub-section.

A general-purpose fractional zoom algorithm has been developed. This algorithm will be implemented, and a series of zoom levels between 1X and 2X will be activated to approximate continuous zooming when software zooms are coordinated with map level selection.

The importance of the MANDATORY/PRIMARY/SECONDARY feature classification will truly come to light once it is possible to move between levels. It is of primary importance that feature display remain consistent when switching from level to level, so as to minimize user disorientation. (This consideration led to the choice of a globally defined feature color table.) There is no guarantee that the new level will have the same set of features as the old; in general, this will not be the case. When a new level is selected, any features that also existed at the old level and were displayed -- i.e., had a type of MANDATORY -- will be displayed at the new level as well. If there are still bitplanes available once these MANDATORY features have been loaded, any feature indicated as MANDATORY in the original database definition for this level will be loaded and displayed. If still more bitplanes are available, PRIMARY features will be loaded (but not immediately displayed). This procedure assumes that when a user changes levels -- an operation which may be transparent to the user during a zoom operation -- all available information relevant to the user's needs will be displayed. Of course, nothing can be done if certain features at the old level are not available at the new level.

It has been evident since the first attempt to digitize maps containing alphanumeric information that the limited resolution of the CRT screen makes all but the largest text fonts illegible. At the Architecture Machine Group of the Massachusetts Institute of Technology, recent work has demonstrated the ability to improve the "virtual resolution" of text by using a two bit greyscaling scheme[2]. In the coming year, the feasibility of incorporating this approach into the VFS terrain map display system will be studied. This would involve reserving two foreground bitplanes which together provide the alphanumerics.

Finally, to more fully demonstrate the utility of the VFS as a map display system, a scenario will be developed which includes overlaid graphics for hypothetical SIGINT data.

### 9.3 LONG-RANGE IMPROVEMENTS

The development of the VFS support software, apart from serving as a satisfactory proof-of-concept for a video disc-based terrain map display system with digitally-processed image manipulation, has brought to light some deficiencies in the VFS architecture. Correcting these problems would involve major revisions to the VFS itself -- if not total reconstruction with a new architecture -- or reconfiguration of the VFS hardware/software environment. The four topics discussed below will not be considered for actual implementation until the work outlined in section 9.2 has been fully completed.

#### 9.3.1 Larger Image Memory

When the VFS was originally designed, memory cost limited the frame store size to 512 x 512. It is now clear that this causes serious problems in the implementation of a smooth translation over a large area; namely, real-time roam is impossible with this architecture. Any point can be brought to the center of the screen by loading sections of four adjacent maps. But it is impractical to load four new maps each time the user wishes to shift the image a single pixel in any direction; it could be done, but certainly not in real time.

With memory costs decreasing, it would now be feasible to build a frame store with a larger image memory, say 2048 x 2048. This would give the user the capability of roaming a screen-sized window around an area 16 times the size of the screen. The database would not have to be accessed until the window approached the edge of image memory.

Figure 11 illustrates how a map load can be handled without losing real-time response. In 11-a, the database is indexed by consecutively numbered vertical strips of maps; strips 1031 through 1034 are shown loaded in image memory. The screen window is moving towards the right, and as it approaches the edge of memory, strip 1035 is loaded in place of 1031 (11-b). Due to virtual memory addressing, the appropriate portions of strips 1034 and 1035, though not physically adjacent in memory, will be displayed properly on the screen.

1A-66,514

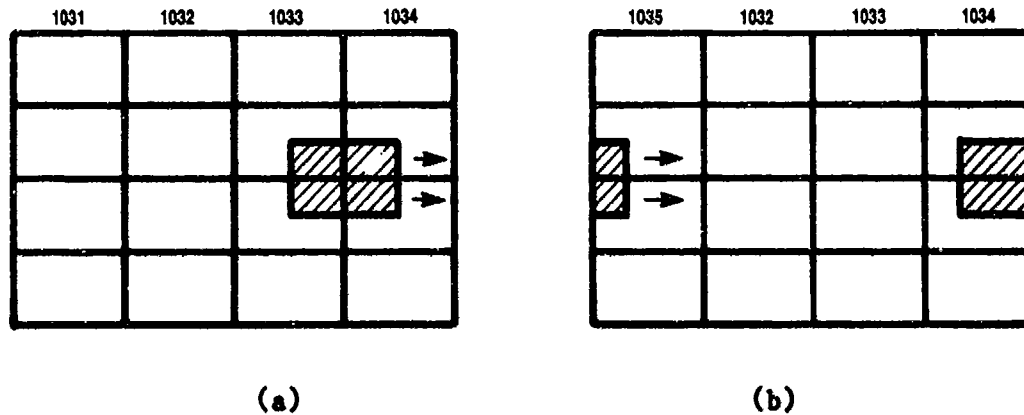


Figure 11

### Large, "Virtual" Image Memory Solves the Translation Problem

The example in Figure 11 considers only horizontal motion and vertically-indexed strips for simplicity. The translations function allows 360 degrees of movement, which can be handled by indexing all 16 blocks of memory, and managing them in a method similar to page-swapping.

The only possibility of additional time delay is at initial system start up time and when switching to a new level, when all 16 blocks must be loaded. This delay can be minimized by first loading the 4 blocks around the display center point, and loading the other 12 after the displayed area has been loaded.

#### 9.3.2 Multiple Video Disca

Loading 4 or 16 maps will intensify a problem which we have already come to notice loading only one map -- namely, that with a series of video disc frames required for a composite map photo, the time required for disc searching becomes a significant factor. It is expected that the delay required for loading new maps will provide unsatisfactory response time. Search time can be cut down

by using two or more video disc players with copies of the same disc. While one disc is fixed over a frame for loading, the other(s) can be searching for other frames.

### 9.3.3 Transparent Microcode Swapping

The planned method of approximating continuous zoom involves swapping the microcode for a variety of zoom factors in and out of microprogram memory, using two program areas. For example, while a 1X zoom (no magnification) program is running at location 100, the code for the 1.25X zoom would be loaded at location 1100. Once loaded, a change in the start address register results in an instantaneous jump to a 1.25 magnification, and the code for a 1.33X zoom can be loaded at location 100 while the program at location 1100 runs. After the switch to the 1.33X program at 100 has been made, a 1.5X program can be loaded at 1100, and so on. The microcode loads very quickly, so that with code for closely spaced zoom factors, a continuous zooming effect can be produced.

A problem in the VFS architecture, however, causes the raster to temporarily lose synchronization when information is written to microprogram memory. This problem must be corrected to achieve smooth transitions between zoom factors.

An alternate solution would be to expand the size of microprogram memory, so that programs for many more zoom factors can remain resident in memory at the same time. This would reduce or eliminate the need for code swapping.

### 9.3.4 User Annotation of Maps

There may be applications for the VFS where the user would want to make annotations associated with a particular map photo; such annotations might include terrain information updates, notes on road passability, bridges missing, etc. Since these annotations refer to the background map rather than to the overlaid intelligence graphics, the VFS should manage the annotations, unburdening the intelligence application package from the task.

The VFS is well suited to such a scratch-pad feature. One or more bitplanes could be reserved for digitized input from a pointing device, tablet, or mouse whereby a user may draw arrows, outline regions, write notes, etc. This information must be stored and associated with the map, so that it can be recalled when the user returns to the particular map photo.

One approach to the storage issue would be to compress the digitized information (via run-length encoding or a similar scheme) -- which would be relatively sparse -- and store it on magnetic disk, maintaining a directory linking annotation frames with map photos.

An alternate method which does not require such large amounts of digital storage space involves storing the frames on magnetic "video disks" in NTSC format. These read/write devices are commercially available and are designed for still framing in video applications. The capacity is much lower than optical video discs -- typically a few hundred frames -- but it is sufficient to record the number of frames which would probably be annotated.

#### 9.4 SURVEY OF COMMERCIALLY AVAILABLE SYSTEMS

When the VFS was originally designed and built in 1979, it was a hybrid device representing the state of the art in real-time digitization, image processing, and frame buffer technology. Experience has shown only one significant drawback in the design, as discussed in the last section: image memory size. The limiting factor was not the technology but the cost, which has since decreased significantly.

Three years ago the VFS was far ahead of commercially available systems in terms of memory size (its 24-bit pixel depth) and image processing capabilities (notably the fractional zoom programmability). Commercial vendors have caught up in the interim, and, in some cases, overtaken the VFS in processing power and sheer memory size. If a real need arises to enhance (i.e., redesign) the VFS along the lines discussed in the previous section, it may prove more feasible or more cost-effective to adapt an off-the-shelf device for use in a terrain map display system. This section is a general survey of some commercially available products, focusing on their ability to match the capabilities of the VFS -- the currently implemented functions as well as the proposed enhancements.

##### 9.4.1 Requirements of an Off-the-Shelf Device

Certain features of the VFS have proven to be integral to the device's use as a terrain map display processor, while other specifications allow some flexibility which will not adversely affect overall system performance.

The need for at least 24 bitplanes should be considered an inflexible requirement. The current VFS architecture permits 12 foreground features, leaving four bits per color for the background, providing an excellent color background image. It would be unreasonable to give up any of the 12 foreground bitplanes; as the number of map features decreases, so does the need to selectively remove individual features from the display -- the major premise of the VFS in the first place. So, a suitable off-the-shelf device must have at least 24 bitplanes.

The only real constraint on image memory size is that it be an integer multiple of  $512 \times 512$ . This is necessary because video discs store frames in standard NTSC (525-line) format. A  $512 \times 512$  image memory is the most direct implementation; each memory image requires one video disc map (that is, one series of frames). But decreasing memory costs encourage larger image memory --  $1024 \times 1024$ ,  $2048 \times 2048$ , etc. The absolute minimum, however, is  $512 \times 512$ .

With 4 bits per color for the background image,  $2^{12}$  or 4096 distinct colors are possible. A more crucial figure, however, is the number of foreground colors. The VFS color lookup table is a 12-bit in, 9-bit out table. (The 12 input bits are the 12 bitplanes.) The 9 output bits code 3 bits per color for intensity level input to the D/A convertors, producing  $2^9$  or 512 simultaneously displayable colors. Many available systems boast 16 or 64 colors "displayable from a palette of over 16 million." This corresponds to a lookup table with 4 or 6 bits in, and 24 bits out (8 bits per color). The VFS's 512 colors are much more than are needed to differentiate between the features and intersections displayed at any given time. 16 colors, however, may not be sufficient; with 12 planes for foreground features, only 4 distinct colors remain for intersection specification. 64 active colors should be sufficient, resulting in a 12-bit in, 6-bit out color lookup table.

For real-time video frame digitization, the VFS has three 6-bit A/D convertors. With a background/foreground allocation allowing the maximum of 12 bitplanes for foreground features, only 4 of the 6 available A/D convertor bits are used for background frame digitization, and this provides sufficient quality color background images.

Finally, the Video Processing Element is based on a  $2 \times 2$  pixel convolver for image processing functions; the convolver is used mainly for zooming operations. This small array results in locally defined interpolation; a larger convolver reduces localization and gives a smoother zoomed image. A  $3 \times 3$  array would be an improvement; programmable  $N \times M$  would be ideal. An investigation

into the image quality/response time tradeoffs would have to be conducted, and may reveal that the power of an  $N \times M$  convolver could be wasted in an environment with real time or near real time response requirements.

The products surveyed fall into three broad device classes: image processors, graphic display controllers, and individual boards. Except in the few cases noted, the devices fail to meet the requirements stated above.

#### 9.4.2 Image Processors

All of the image processing devices surveyed have A/D convertors for video input, and all meet the memory size requirements, at least at first glance. However, the memory planes are usually configured so that they may be used only for full color digitizations, or "pseudo-coloring" as a tool for image enhancement. Some number of separate "graphic overlay" planes are provided; these bitplanes are color-mapped, but are designed to display digitally generated/transmitted images and do not accept input from the A/D convertors. All of the devices provide pixel-step translation and integer zooming, in either consecutive steps or powers of 2, as well as the usual image processing and enhancement features (image addition, averaging, etc.).

Comtal Corporation's Vision One/20 Model M10 is the Cadillac of image processing systems. Expandable to 16  $512 \times 512 \times 8$  images, this stand-alone, dual-user system offers four graphic overlays, programmable cursor shape, and a programmable video processor. An optional continuous zoom facility is available, which permits fractional zooming in .002X steps up to 512X. The cost for a basic  $512 \times 512 \times 24$  system is over \$60,000; add-ons bring the price substantially higher.

Comtal also sells a series of host-driven models. The 8000-R Model 30 has a maximum configuration of 3  $512 \times 512 \times 8$  images, with 4 graphic overlays and a programmable cursor. The Model 30 is available with maximum memory for under \$35,000.

Grinnel Systems manufactures a full line of frame buffers, graphic display systems, and image processing systems. The GMR 273 "Image Frame Buffer" has a maximum  $512 \times 512 \times 24$  configuration, with 4 graphic overlays and 4 programmable  $15 \times 15$  cursors. Two of the overlay bits are also fed to the most significant bits of the video lookup tables (three tables, each 10 bits  $\times$  8 bits).



International Imaging Systems' Model 70 Image Computer can provide up to 12 512 x 512 x 8 images. A hardware split screen feature allows an arbitrary window on a 2048 x 1536 (x 8 bit) virtual memory. 8 graphic overlay planes are provided. Only the overlay planes are color-mapped; the 8-bit in, 16-bit out Color Assignment Function Memory provides 5 output bits for each color channel, with the 16th bit selecting the mode for combining graphic data with image memory: additive or overlay (replace). The cursor is programmable, 64 x 64 pixels.

The VICOM Digital Image Processor by Vicom Systems, Inc. may contain 4 images, each 1024 x 1024 x 16. Four graphic overlays, a programmable cursor, and a programmable 3 x 3 pixel convolver are included.

#### 9.4.3 Graphic Display Controllers/Terminals

This group of products includes devices ranging from raster graphic controllers to self-contained graphic terminals. The significant difference between these devices and image processors is that they were designed for digital picture generation, rather than processing of video images. As a result, most of these devices do not include A/D convertors for real-time video input. They do provide a DMA interface for fast transfer of digital data, making it possible to use external A/D convertors. The lack of a real-time frame grabber, however, may make these items unsuitable for this application.

Genisco Computer Corporation's products include the G-6100 series Interactive Graphic Terminal and the GCT-3000 raster display system. The G-6100 line features 1536 x 1024 resolution, but with only 4 memory planes (offering 16 colors from a selection of only 64). The G-6110 has a viewable area of 1296 x 1024; the G-6120's window is 768 x 512. Two graphic overlay planes are available as an option.

These devices are representative of many products on the market with a set of features simply too divergent from VFS project requirements. They are high resolution, powerful display processors with a full complement of graphic generation/manipulation capabilities, often with ACM CORE Standard software. In a mapping application with a totally digital database, such devices would be well-suited; for the present video disc database, they simply do not have enough memory planes.

Genisco's GCT-3000, on the other hand, provides a maximum resolution of 1280 x 1024, with 24 memory planes. Optional modules provide character/vector generation and scroll/zoom functions (1, 2, 4, or 8X magnification). Genisco has a standard DMA interface available for the Perkin-Elmer (Interdata) 8/32 minicomputer.

The RDS-3000 by Ikonas Graphics Systems is an ambitious graphic controller with features comparable to the image processing systems mentioned above. In fact, when the new Video Rate Processing Module is made available, the RDS-3000 will have the capabilities to perform image addition and subtraction, averaging, filtering, and convolution. An optional real-time video digitizer is already available.

The RDS-3000 is built around an expandable, modular concept, allowing a maximum configuration of 2048 x 2048 x 24, with a 1280 x 1024 window. There are 8 graphic overlay planes and one cursor (32 x 32 pixels). The overlay planes, together with the cursor plane and a control register bit, provide 10 bits of input to the color lookup tables. The 3 tables each produce 8 bits of intensity information, with an option for 10 output bits feeding 3 10-bit D/A converters, making the RDS-3000 the only commercially available system we have yet encountered which can provide over one billion color hues. Of course, as a graphic processor, the RDS-3000 can generate vectors, polygons, text, and has a hardware polygon fill. A 16 x 16 user-microprogrammable fast matrix multiplier provides 2-D and 3-D translation, rotation, and scaling. The optional Multiperipheral Controller (MPC) is a Motorola 68000-based module which allows direct control of peripherals from the RDS-3000; Ikonas plans to add the UNIX operating system to the MPC, making it a stand-alone host.

Jupiter Systems, Inc. manufactures a color graphic terminal which resides in the middle ground between the Genisco G-6100 type devices and the Ikonas RDS-3000, in terms of the VFS project requirements met. The Jupiter 7 offers high resolution (1024 x 1024 with a 768 x 575 window), but only 8 memory planes. 256 colors are selectable from 16.7 million, adequate for feature discrimination. The Jupiter 7 has read and write mask registers for selective bitplane control, hardware blink with user-programmable delay rates, and a hardware blue-line grid (a feature particularly useful in CAD/CAM applications). Character, vector, circle, and filled polygon generation are included, as well as translation and zoom, in integer steps from 1X to 16X. An excellent feature is the hardware

anti-aliasing of vectors, which eliminates the "jaggies" resulting from the display of diagonal lines on a raster device. The new AED767 by Advanced Electronic Design, Inc. matches the Jupiter 7's features point for point; Jupiter's founders were the original designers of the AED512. Even with only 8 memory planes, the Jupiter 7 is probably the least expensive off-the-shelf device (under \$16,000) which can provide an environment for extended investigations into terrain map displays. Of course, the absence of a frame grabber remains a major obstacle.

The Lexidata Corporation System 3400 Display Processor is a peripheral device which performs picture display and image manipulation functions for a host minicomputer; keyboard, monitor, etc. are not included. Standard configurations include 512 x 512 x 16 and 1024 x 1024 x 8 with a 640 x 512 window. Up to 5 overlay memory planes can be added. The 3 color lookup tables are 12-bit in, 8-bit out, providing 4096 simultaneously displayable colors from a palette of 16.7 million. A programmable 64 x 64 cursor is included, as well as a full-screen cross-hair cursor. Lexidata provides a standard DMA interface to a Perkin-Elmer 8/32 minicomputer.

#### 9.4.4 Modular Boards

An approach which may prove to be the most feasible for acquiring a VFS-like device on the commercial marketplace is to use modular boards. Two vendors were surveyed; the products offer very different capabilities at an equally different price.

Graphic Strategies, Inc. manufactures a board which is plug-compatible with the Motorola MC68000 VERSAbus and VERSAmodule\* systems, called the VERSAgraphic Module. Each board is available in 512 x 512 or 1024 x 1024 resolution, with 4 bitplanes. One example configuration mentioned in the company literature uses one VERSAgraphic module for background map display, and one for overlay data. Only 16 colors are available from a palette of only 64. This is basically a graphic controller board. Standard and high quality text are generated, as well as graphic primitives (including polygon fill). Integer zoom in steps from 1X to 16X is featured, but panning and scrolling can only be accomplished in 16 pixel increments. There is also a user-programmable cursor. Image

---

\* VERSAbus and VERSAmodule are registered trademarks of Motorola Semiconductor Products, Inc., Phoenix, Arizona

storage and retrieval is supported with run-length-encoded data compression. Single-unit prices are \$2600 for 512 x 512 and \$4700 for 1024 x 1024, with substantial quantity discounts, up to 30% off for quantities of 100 or more. A single evaluation unit may be purchased at the 30% discount rate (\$1820 or \$3290, depending on resolution).

Imaging Technology, Inc. manufactures a series of boards which may be combined into a full image processing system, the IP-512. The modules include the FB-512 Frame Buffer, the AP-512 Analog Processor, and the ALU-512 Arithmetic Logic Unit.

The individual 512 x 512 x 8 Frame Buffer boards can be combined to form larger and deeper image memories. The literature gives the maximum configuration as 4096 x 4096 x 16, but the company has informed us that a 2048 x 2048 x 24 configuration is also possible (requiring 48 FB-512 boards). The memory modules contain the logic for pixel scroll, pan, write mask and 2X zoom functions. Bitplane 0 may act as a pixel write protect mask for the other 7 bitplanes.

The AP-512 Analog Processor is available with 6-bit or 8-bit A/D convertors, and with 1 or 3 D/A convertors. Lookup tables are provided to give 256 colors from a selection of 16.7 million. For a 24 bit pixel depth, 3 AP-512 boards would be required.

The Arithmetic Logic Unit is a high speed pipeline processor, capable of operating with four FB-512 Frame Buffer channels and a single AP-512 Analog Processor channel, in parallel, at a rate of 10 million pixels per second. The usual image enhancement functions are available -- summation, averaging, exponentially weighted averaging over N frames (N = power of 2 up to 128) and frame subtraction. The ALU-512 is also capable of performing convolutions with an N x N kernel.

Each 8-bitplane FB-512 board has a list price of about \$4000; each AP-512 Analog Processor with 6-bit A/D convertors and 3 D/A convertors is \$3200, and each ALU-512 is \$3,000. Thus, a fully configured 2048 x 2048 x 24 system would cost approximately \$300,000. The company has indicated that a discount of approximately 30% would be applied to such an order, due to the large number of boards involved; we could also expect additional discounts for multiple units.

The original materials budget for the Video Frame Store, prepared in February, 1979, listed the price of a single 64K RAM chip at \$125. That chip now sells for \$7.25. The 32 chips required for a single FB-512 (512 x 512 x 8) thus have a cost of \$232;

Imaging Technology is undoubtedly trying to recover its development costs by pricing the board at \$4000. As memory costs continue to decrease, and as competitors of Imaging Technology emerge, we can expect to see substantial decreases in the prices of modular frame buffer boards, and the analog and image processing boards as well.

#### LIST OF REFERENCES

1. David H. Lehman and David C. Pasterchik, "Digital Processing of Videodisc Map Imagery," 3rd Annual Conference on Videodiscs, Society for Applied Learning Technology, August 82.
2. Christopher Schmandt, "Soft Typography," Information Processing 80, S. H. Lovington, ed.
3. F. B. Pope, "A Digital Video Frame Store/Graphic Processor," WP-22139, February 1979.

## APPENDIX A

### VFS EXECUTIVE USER'S MANUAL

#### A.1 INTRODUCTION

The purpose of this manual is to collect, in a definitive source and reference guide, all information relating to the operation of the VFS Executive program.

Since the users of the Executive are VFS project members who are thoroughly familiar with the VFS architecture and hardware/software environment, no attempt is made here to provide a context or to explain the commands in detail. This manual does provide, however, a detailed description of syntax, descriptions of all commands, and explanations of any error messages which may occur.

#### A.2 BASICS

##### A.2.1 Invoking the VFS Executive

The VFS Executive program is located in the user directory for the "vfs" account; the program name is also "vfs". The easiest way to start the Executive is to login as "vfs" and invoke the program. The Executive will respond with a header and a command prompt:

```
login: vfs
% vfs
```

```
-----
Video Frame Store Executive                VFS/UNIX - version 1
-----
VFS>
```

##### A.2.2 Line and Loop Commands

Some of the commands, called "line" commands, accept all of their arguments on the line when the command is invoked by name. The command is executed, and the "VFS>" prompt is displayed for a new command.

"Loop" commands, on the other hand, allow their functions to be applied repeatedly, with different arguments. Such commands present their own prompts, consisting of the command name followed by a right angle bracket (e.g. "SET>", "DUMP>").

### A.2.3 Exiting

The way to exit from a loop command, or from the Executive itself, is to press "ctrl-d". It is also possible to abort any Executive command by issuing an interrupt ("DEL" key). The VFS Executive header is displayed to remind you that you are back at the top-most command level, and the "VFS>" prompt is issued for a new command.

## A.3 COMMAND ARGUMENTS

### A.3.1 Syntax

Command arguments have the same syntax, whether for a line command or for a loop command.

In cases where multiple arguments are required, the arguments are separated by commas. For line commands, there must be a comma (not a space) between the command name and the first argument.

Each argument consists of a value -- which may be a file name, memory name, etc., depending on the command -- followed by a list of argument qualifiers called "switches".

A switch may indicate an address, a pair of line or pixel values, a bitplane mask, etc. In specifying a switch, the switch name is preceded by a backslash (\). Immediately following the switch name, the value(s) are specified, enclosed in square brackets and separated by commas. All numeric values are in octal.

For example, the argument specification

```
rf\x[100,200]\y[20,40]\m[37]
```

indicates the argument rf (for "red frame store memory"), with switches indicating pixels 100 to 200, lines 20 to 40, and bitplane mask 37. (All switches and values will be explained shortly.)

If a command contains so many arguments and switches that it is too long to fit on one line, the special switch \c may be used with any argument to indicate that the command is continued on the next line.

The special switch \ex is used in loop commands, especially when run from an indirect command file (explained in section A.6).



The \ex switch may be used with any argument; it causes an exit from the command input loop after the current line -- which may contain multiple arguments -- has been processed.

### A.3.2 Memory Name Arguments

Certain commands require memory names as arguments, with switches specifying address or pixel ranges, bitplanes, etc. The memory name abbreviations recognized by the Executive are as follows:

mi	microprogram memory
ma	color mapping memory
co	coefficient memory
ri	red intensity mapping memory
gi	green intensity mapping memory
bi	blue intensity mapping memory
rf	red frame store memory
gf	green frame store memory
bf	blue frame store memory

### A.3.3 Memory Specification Switches

For commands which operate on a range of any of the VFS memories, specific switches are used with the memory name abbreviation. The CLEAR, SET, and DUMP commands are loop commands which accept the following switches on memory name arguments:

\all	entire memory
\r[first,last]	memory address range
\x[first,last]	pixel range
\y[first,last]	line range
\m[mask]	bitplane mask

The switches \x, \y, and \m are for frame store memories only.

If no switch is given with a memory name, \all is used.

The loop commands DRAW and ERASE always use a mask switch (\m) and a pair of switches specifying endpoints:

\p0[line,pixel]	starting line and pixel
\p1[line,pixel]	ending line and pixel

#### A.3.4 Global Switches

Sometimes a command accepts a switch but no argument; such a switch is called a "global" switch and is applied to the command name the way normal switches are applied to arguments. For example:

```
VFS>clear\all
```

will clear all VFS memories (and will not result in a command input loop for the CLEAR command).

#### A.4 INDIRECT COMMAND FILES

A facility is provided in the Executive whereby sequences of commands may be stored in a file and called up for later execution. Once a file of commands has been created (using the UNIX\* text editor), it may be called up from the Executive by typing the character "<" followed by the file name. This redirection of input may occur at any stage in the Executive (i.e., even within a command input loop). Once an indirect command file has been activated, all input is taken from that file. If an error occurs, the file is closed and normal keyboard input resumes. Likewise, when the end of the file is reached, normal keyboard input resumes. A file may include an activation of another indirect command file, but when the second file is completed, the keyboard is activated; the first file is not continued from the point where it opened the second. Files may be arbitrarily chained in this way. It is convenient to think of the first indirect command file activation as a subroutine call (since it "returns"), and each successive link in the chain as an unconditional jump.

Comments may be inserted in an indirect command file by typing a line beginning with a semicolon (";"). When the line is encountered by the Executive, it will be echoed to the terminal screen. This feature is useful in providing status messages and interacts quite well with the PAUSE command (paragraph A.6.2 below).

## A.5 UNIX COMMAND EXECUTION

When a line is preceded by the character "!", it is passed directly to the shell for interpretation as a UNIX command. Any operation may be performed; for example, the editor may be invoked to create or change an indirect command file.

## A.6 COMMANDS

For quick reference, the commands are listed in Table A-1. The table gives the command type (line or loop) and the argument(s) expected. For full command syntax, argument, and switch descriptions, however, refer to the text in this section. The VID command is a special command with its own input handler (neither line nor loop) and is described in section A.7. All commands may be abbreviated to the first two characters of the command name.

<u>Command</u>	<u>Type</u>	<u>Argument(s)</u>
help	line	topic
pause	line	---
run	line	start address (global switch \s)
halt	line	---
reset	line	---
clear	loop	memory specification
set	loop	memory specification
load	line	file name
dump	line/loop	destination/memory specification
draw	loop	endpoints, mask
erase	loop	endpoints, mask
vid	*special*	---

Table A-1

Commands and the Arguments They Expect

#### A.6.1 HELP Command

HELP is a line command which takes a topic name as its argument. It displays up to a screen full of information on the topic, with cross-references to related topics. If no topic is specified, general syntax is explained, along with a list of topics. The topics available include all commands (use the command name as the argument) as well as "memory", "switch", "indirect", "comments", and "unix".

#### A.6.2 PAUSE Command

When sequences of commands are executed from indirect command files, it is sometimes necessary to temporarily halt at a certain point and to continue after a short time. The PAUSE command waits for the RETURN key to be pressed, then allows the command file to continue.

#### A.6.3 RUN Command

To start the VFS from a particular location in microprogram memory, use the RUN command with a global switch for the start address; e.g.

```
VFS>run\s[100]
```

If the VFS had been running and was halted, it may be restarted from the old address by issuing the RUN command with no start address switch.

#### A.6.4 HALT Command

This command causes the VFS microsequencer to halt. Since image memory refresh is handled by a software subroutine in microprogram memory, the frame store memories will begin to decay when the VFS is halted.

#### A.6.5 RESET Command

If video sync is lost, the RESET command will restore it.

#### A.6.6 CLEAR Command

The CLEAR command is a loop command which clears to zero specified ranges of any of the VFS memories. Sections A.3.2 and A.3.3 describe the memory name abbreviations and the switches for specifying memory ranges.

#### A.6.7 SET Command

The SET command is a loop command which sets to one all bits in specified ranges of any of the VFS memories. Sections A.3.2 and A.3.3 describe the memory name abbreviations and the switches for specifying memory ranges.

#### A.6.8 LOAD Command

LOAD is a line command which is used to load data files into VFS memories. Files must be in a special format; all files created by the DUMP command and by the VFS Assembler have this format. The file itself contains certain header information which indicates where in the VFS memories the data should be placed.

More than one file may be loaded with a single invocation of the LOAD command, by specifying an argument list of file names. Any combination of the following three switches may be applied to each individual file:

\zero	clear <u>all</u> memory before loading the file
\run	after loading, run from the start address indicated in the file
\origin[address]	(applies only to assembler object files) use the indicated address as the origin for the object code, overriding any default origin in the file

#### A.6.9 DUMP Command

The DUMP command uses line arguments to specify the destination of the dump, and then uses an input loop to accept memory range specifications. On the command line, a combination of switches and a dump file name are entered. Three global switches may be used: the \all switch indicates that the entire VFS memory should be dumped, and the other two global switches specify a dump destination:

<code>\printer</code>	dump to the printer
<code>\terminal</code>	dump to the terminal screen

Only one of these two switches should be used; if both are used, the dump is to the printer only.

In addition to the destination for a formatted dump, the data may be written to a dump file; the file name is used as the argument. It is thus possible to dump both to a file and to the printer or terminal with a single DUMP command. For example:

```
VFS>dump\all\printer,bigfile
```

will produce a hardcopy output listing of the entire VFS memory, as well as a file called "bigfile" which may later be loaded by the LOAD command.

If no destination switch and no file name argument are specified, the dump is to the terminal by default.

DUMP uses a command input loop to prompt for memory ranges to be dumped. This permits ranges of different memories to be dumped to one file. (For example, an image environment may be saved by dumping frame store memories and color mapping memory.) See sections A.3.2 and A.3.3 for descriptions of the memory names and switches for memory ranges.

#### A.6.10 DRAW Command

Lines may be drawn into specific frame store memory planes using the DRAW command. This is strictly a loop command which accepts a frame store memory name (rf, gf, or bf) as an argument with the following switches:

<code>\m[mask]</code>	mask for specific bitplane(s) to be written
<code>\p0[line,pixel]</code>	starting point
<code>\p1[line,pixel]</code>	ending point

#### A.6.11 ERASE Command

This command is the complement of the DRAW command, and uses the same arguments and switches. Rather than drawing a line by setting locations in the specified bitplanes to 1, it erases a line by setting the locations to 0.

## A.7 THE VFS INTERACTIVE DEBUGGER (VID)

An interactive debugger is provided for the VFS, called VID. VID is invoked as an Executive command (command name: "vid").

The primary use for VID is examining and modifying memory locations, and it has its own command syntax specifically tailored to this purpose. The CRT screen is split to display the contents of certain frequently used memory locations in "windows" on the top portion of the screen, with the bottom few lines set up as a scrolling area where commands are entered. Whenever a register with a window is changed by a command entered in the scrolling area, the value is updated in the window. Figure A-1 shows the screen layout of the register windows.

### Video Frame Store Interactive Debugger

	RED	GRN	BLU	
Read Mask	###	###	###	Intensity Map ##
Write Mask	###	###	###	Grey Scale #
Lower Limit	##	##	##	A/D Mode #
Upper Limit	##	##	##	

LINEs	PIXELS	Start Address
#	###	###

Figure A-1

### VID Screen Windows for Common Registers

VID operates with the terminal in raw input mode; characters are read as they are typed, instead of being transmitted in an entire line when the RETURN key is pressed. Thus, a number of different characters are defined as "break" keys which cause some operation to be performed. Any command arguments are entered before the break character, so that the operation may occur immediately after the break character is entered.

### A.7.1 Arguments

All commands in VID accept either 0, 1, or 2 arguments, which precede the break character. When 2 are required, they are separated by commas. Arguments usually supply addresses or data, and are interpreted as octal by default. Any number which is preceded by the character "#" is interpreted as decimal.

Addresses are memory word addresses. The size of a word depends on the particular memory. Microprogram memory word size is 16 bytes, coefficient memory is 4 bytes, and all others are 1 byte. Individual bytes may be addressed by giving the byte number in parentheses immediately after the word address.

When an address argument is specified, the associated memory must be indicated. VID accepts the same memory name abbreviations as the Executive (listed under paragraph A.3.2 above), with the addition of "A" and "I".

The memory name "A" stands for "absolute memory" and is used mainly to address certain special control registers (A/D limits, read and write masks, etc.) although any address in VFS memory may be addressed in this manner if its absolute address is known.

Twelve sequentially numbered index registers are provided for addressing. They are used with an address by typing a semicolon (";") and the register number immediately after the address. A register may be accessed or set by using the special memory abbreviation "I", with the index register number (between 0 and 11).

Frame store memory locations are accessed with line and pixel values, rather than word and byte values. The line number is given first, followed by the pixel number in parentheses.

For convenience, special two-character mnemonics are provided for certain common control register locations. When using a mnemonic instead of the longer version of the address, the mnemonic must be preceded by the character "@". For example, the address for the red read register, A177770, may be replaced by the more terse version @RR. The mnemonics, and the addresses they represent, are listed in table A-2.

### A.7.2 Break Characters

Table A-3 is a quick reference to the break characters and their functions. Each is described in more detail in the paragraphs which follow.



<u>Mnemonic</u>	<u>Address</u>	<u>Description</u>
AD	177740	A/D convertor mode
RB	177741	red lower A/D limit
RT	177742	red upper A/D limit
GB	177743	green lower A/D limit
GT	177744	green upper A/D limit
BB	177745	blue lower A/D limit
BT	177746	blue upper A/D limit
RW	177760	red write mask
GW	177761	green write mask
BW	177762	blue write mask
T1	177764	translation/blanking bit registers
T2	177765	
T3	177766	
RR	177770	red read mask
GR	177771	green read mask
BR	177772	blue read mask
GY	177773	grey scale (background/foreground alloc.)
IM	166000	intensity map modes
DR	177767	frame store data register

Table A-2

#### VID Register Mnemonics

<u>Argument(s)</u>	<u>Character</u>	<u>Effect</u>
address	/	display contents of location
value	RETURN	close location (write new value)
value	LINE FEED	next byte " "
value	-	previous byte " "
value	N	next word " "
value	P	previous word " "
value	L	list entire word " "
address	E	execute
	H	halt
	U	reset
line, pixel	X	translate
#lines, #pixels	S	slide
frame #	Z	search video disc
	Q or ctrl-d	quit
	F1	jump into windows
	?	pause

Table A-3

#### VID Break Characters

A.7.2.1 "/". Typing a slash causes the contents of the currently open location to be displayed. If an address precedes the slash, that address is opened and its contents displayed.

A.7.2.2 "RETURN". The currently open location is closed. If a value precedes the RETURN key, that value is written to the location.

A.7.2.3 "LINE FEED". The current location is closed, and the next sequential byte is opened. If a value precedes the LINE FEED key, that value is written to the currently location before it is closed.

A.7.2.4 "^". The current location is closed, and the previous sequential byte is opened. If a value is given, it is written to the current location before it is closed.

A.7.2.5 "N". The current location is closed, and the next sequential word is opened. If a value is given, it is written to the current location before it is closed.

A.7.2.6 "P". The current location is closed, and the previous sequential word is opened. If a value is given, it is written to the current location before it is closed.

A.7.2.7 "L". List entire word; all bytes of the word containing the current location are displayed. Word size depends on the memory; for micro memory it is 16 bytes, for coefficient memory it is 4 bytes, for color mapping memory 2 bytes, and 1 byte per word for all others.

A.7.2.8 "E". Begin execution at the current start address, or if an address is specified, first write it to the start address register.

A.7.2.9 "X". Halt the VFS.

A.7.2.10 "U". Reset the microsequencer (restores sync).

A.7.2.11 "X". Translate the displayed image to the absolute line and pixel offset specified by the two values which precede the X. The values may be negative.

A.7.2.12 "S". Slide the displayed image a relative number of lines and pixels, positive or negative, as specified by the arguments.

A.7.2.13 "Z". Search the video disc for the indicated frame number. Since frame numbers are in decimal, the argument to the Z command is customarily preceded by the character "#" to indicate a decimal argument.

A.7.2.14 "Q" or "ctrl-d". Exit VID and return to the top command level of the Executive.

A.7.2.15 "F1". This character causes a special mode to be entered in which any of the registers displayed in screen windows may be changed by positioning the cursor at the start of the appropriate window and typing the new value. The cursor may be moved left, right, up, and down among the windows using the keys F11, F12, F15, and F16, respectively. "Window input mode" is terminated by pressing "ctrl-d", at which time all changes actually take effect and the cursor is returned to the scrolling area at the bottom of the screen.

A.7.2.16 "?". Pause until a key (any key) at the keyboard is pressed.

### A.7.3 Indirect Command Files

The VFS Executive feature of taking input from an indirect command file is included in VID. Indirect command files are activated by typing the character "<" followed by the file name.

### A.7.4 UNIX Command Execution

UNIX commands may be executed from VID by preceding the command with the character "!".

## A.8 EXECUTIVE ERROR MESSAGES

This section lists any error messages which may result, grouped according to the context in which they occur. Most of the messages are self-explanatory; a few are briefly elaborated on here.

#### A.8.1 Top Level Command Entry

Unrecognized command

Illegal use of global switches

The switch parsing routine includes an allowance for global switches with no command name; the Executive, however, does not use this feature.

#### A.8.2 Command Input Loop Arguments

Unrecognized memory specifier

Unrecognized switch

Illegal memory/switch combination

Certain switches are illegal with certain memories, e.g. the switches \x, \y, and \m may only be used with frame store memories.

Too many switches

There may be 10 on each argument.

Incorrect number of values for switch

Switch value must be numeric

Switch values specify illegal memory limits

Magnitude of first value exceeds second

For switches with 2 address values, the second may not be less than the first.

Transfer address out of range

The value of the \s switch must fall between the bounds of microprogram memory (0 - 1777).

#### A.8.3 HELP Command Topic Name

Unable to access help text file

#### A.8.4 RUN Command Transfer Address

Illegal switch

Too many switches

Missing switch value

Transfer address out of range

Transfer address is undefined

If the RUN command is used without an \s switch before the VFS was ever started, there is no "current start address".

#### A.8.5 CLEAR and SET Command Line Arguments

Illegal switch

#### A.8.6 LOAD Command Line Arguments

Illegal or unrecognized switch

File not found

Input file not in load format

Checksum error detected -- loading continues

With each block of data, a checksum byte is dumped for loader verification. If an error is detected at load time, it is reported but does not halt the load.

EOF detected before logical EOF

An end-of-file was reached while attempting to read a block of data. The file is probably damaged.

Origin address out of range

The value of the \origin switch, for relocating assembler programs, must fall within microprogram memory bounds.

#### Program crosses upper memory range

The program being loaded has run into the upper bound of microprogram memory.

#### Relocated address out of range

A relative address used by the program, when relocated according to the new origin, is not within the bounds of microprogram memory.

#### A.8.7 DUMP Command Line Arguments

Unrecognized or illegal switch

Can't open dump file

Can't access line printer

#### A.8.8 VID Messages

Command buffer full

Too many characters for VID to handle. This message should never appear because the command buffer allocated is large.

Unrecognized character

Numeric overflow

Unrecognized memory specifier

Word address out of range

Word addresses given are relative to the start of the particular memory being addressed; thus, each memory's addressable space starts at 0. The upper limit depends on the particular memory.

Illegal syntax

Remember, arguments precede commands in VID.

Byte address out of range

### Illegal index register

Registers are numbered 0 through 11.

### Command decoding error

### Location not open

Attempt to write data when no location is open. Use a slash ("\") to open, then write the data.

### Command/argument mismatch

### Nested indexing not allowed

You can't use index registers when opening index registers.

### Address/index register memory mismatch

### Transfer address undefined

### Can't list frame store memory or index registers

The list command ("L") is restricted to those memories which are organized in bytes and words. Frame store memories are divided into lines and pixels.

### Unrecognized mnemonic

## APPENDIX B

### APPLICATION DEFINITION USER'S MANUAL

#### B.1 INTRODUCTION

This manual is a guide to the definition of a database for a VFS map display application. It essentially traces the execution of an interactive database definition program. The most effective way to use this manual is to have it on hand at the terminal while defining an application database. First-time readers of the manual, who have never used the definition program, are referred to sections 4 and 5 of the main body of this paper, for details on the database organization and a general introduction to the definition package.

It is assumed that the users of this manual are thoroughly familiar with the VFS architecture and generally aware of the database format. For the present, VFS project members are the sole users of the database definition routine. Ultimately, databases will be defined by separate application developers with little or no knowledge of the VFS architecture. At that time, a more verbose manual will be prepared.

Section B.2 gives basic instructions for invoking the programs, and lists all intermediate and final files created. Three files are created in sequence as the data is entered; sections B.3 through B.5 correspond to the information entered for these files (the Video Disc Directory, the Feature Color Table, and the Application Definition File). Finally, section B.6 briefly describes error recovery.

#### B.2 BASICS

The definition and report routines reside in the directory "vfs"; the most convenient way to invoke the programs is to login under the "vfs" account.

The definition program is a two-pass procedure. The arguments used determine the program pass.

##### B.2.1 Definition - Pass 1

The first pass is the actual data entry phase. The definition program, called "define", is invoked with a unique application name as its only argument. For example,



## **%define fuldademo**

The result of pass 1 is the creation of a file which can be changed with the UNIX text editor. The filename is composed of the application name with the suffix ".ed", e.g., "fuldademo.ed". This file contains every data element entered, each on a separate line, with a brief description on the line with the data.

The data entry cycle is explained in sections B.3 through B.5 below.

### **B.2.2 Definition - Pass 2**

In the second pass of the definition program, the intermediate editable file is read and the application database is created. The "define" program is invoked with the unique application name as the first argument, and a single hyphen as the second, e.g.,

## **%define fuldademo -**

The database created consists of three files with names generated from the application name with appropriate suffixes: ".vdd" for the Video Disc Directory, ".fct" for the Feature Color Table, and ".adf" for the Application Definition File.

The motivation for the two pass nature of the definition program is to avoid repetition of the entire data entry cycle when small database changes must be made. The intermediate "-.ed" file can be conveniently edited, and then pass 2 can be re-run to produce a new database.

It is possible to run pass 2 in "verbose" mode. In this mode, all of the prompts from pass 1 appear on the screen, and the data is written to the screen as it is read from the intermediate file. To run in verbose mode, use "-v" as the second argument, rather than "-".

### B.2.3 Database Report

Once a database has been defined, a printed report can be produced for reference purposes. The reporting program is invoked with the application name, e.g.,

`%report fuldademo`

The report is sent to the standard output file (i.e., the terminal by default), but may be "piped" to the off-line-print (opr) program for hardcopy output.

## B.3 DEFINING THE VIDEO DISC DIRECTORY

The first information entered is the number of levels. Once this has been done, each level's data is entered in a screen "panel" containing prompts and data "windows". Initially, all prompts are in half-bright video, and windows are in half-bright inverse video. As each data item is entered, its window is switched to full-bright inverse video and the cursor appears at the start of the window. After the user enters the data and presses the RETURN key, the window is "removed" by displaying the data just entered in normal video. The cursor is then moved to the next data window.

### B.3.1 Map Scale Information

For each level, the first data requested is the scale value. This is the reduction factor when the map is displayed on the screen, e.g. "500,000" for a 1:500,000 scale map. This scale is in general not the same as the scale of the original source paper map, since some reduction or magnification occurs during the photographic process.

In addition to the scale factor, the scale range must be entered. These values define the full range of magnification for which this level will be used. For example, a range of 200,000 - 800,000 indicates that during a zoom operation, this level will be used as long as the effective scale is within the limits of 1:200,000 and 1:800,000. When zooming in (magnifying), this level will be used with a software zoom until the actual scale crosses 1:200,000, when a new level, with maps drawn at greater detail, is chosen.

### B.3.2 Photo Depth

The "depth" of each map photo (that is, the smallest geographical piece of the mapped region) is the total number of frames for each photo on the video disc, a constant value for the entire level. Three values are entered: number of foreground frames per photo, number of black and white background frames per photo, and number of color background frames. Both black and white as well as color frames may be present on the disc, although only one set will be used when the application is run. The number of color frames is restricted to 1 for a full color frame or 3 for individual color separations.

### B.3.3 Photo Size

Each level is divided into horizontal strips of equal height, and each strip is divided into equal-sized photos. The photo size is determined by two parameters entered at this time: distance between strips, and distance between photos. Each is entered as a triplet of values: degrees, minutes, and seconds. The three windows are activated simultaneously, and the user can move the cursor among them to make corrections by pressing an appropriate function key for left (key F11) and right (key F12) movement. When the user presses the RETURN key, all three values are accepted at once.

### B.3.4 Strip Latitudes

The next piece of data entered is the latitude of the first strip. This consists of four values -- degrees, minutes, seconds, and orientation (a single character, "N" or "S") -- which are entered "simultaneously" in the manner described in the last paragraph. Then, the number of strips is entered. It is unnecessary to enter the latitude of each strip, since it can be calculated from the strip number, the first strip latitude, and the strip height.

Figure B-1 shows the screen input panel just before the number of strips is entered.

### B.3.5 Strip Input Loop

Some information is needed for each individual strip, however, and this data is entered in an area in the lower portion of the screen, which is re-formatted for each strip.

```

Map Level: 1 of 2
---Scale Value 50000      Scale Range: 30000 - 50000

Number of foreground frames per photo      14
Number of black and white background frames 2
Number of color background frames          1

                                DEG MI SE
Distance between strips  0 4 0
Distance between photos  0 5 0

Latitude of first strip  36 12 0 N (N/S)      Number of strips  2

```

Figure B-1

The longitude of the strip is first entered, as a quadruple of values: degrees, minutes, seconds, and orientation ("E" or "W"). Then, the number of photos is entered. Each photo's longitude can be calculated from the photo number, the strip longitude, and the photo width. Finally, the frame numbers on the video disc of the strip's first foreground frame and first color background frame are entered. (The black and white background frames immediately follow the foreground frames for each photo.)

Figure B-2 shows the screen input panel while the data for one of the strips is being entered.

```

Map Level 1 of 2

Scale Value 50000      Scale Range 30000 - 50000

Number of foreground frames per photo      14
Number of black and white background frames 2
Number of color background frames          1

                                DEG MI SE
Distance between strips  0 4 0
Distance between photos  0 5 0

Latitude of first strip  36 12 0 N (N/S)   Number of strips 3
-----
Longitude of strip 2    105 3 0 E (E/W)    Number of photos 3
First foreground frame no 2464 ° First color background frame no 

```

Figure B-2

#### B.4 DEFINING THE FEATURE COLOR TABLE

Before the screen panel for feature color table information appears, a prompt is displayed for the number of features. Then, for each feature, the screen is formatted with prompts and windows for entry of the feature names and colors.

##### B.4.1 Feature Name

The feature name is a unique name, up to 14 characters in length. If an attempt is made to enter a name that was used before, an appropriate error message is printed on the screen.

#### B.4.2 Feature Color

The feature color is a three digit code representing the intensity levels for blue, green, and red (left to right). Each digit may have a value between 0 and 7. Beware of omitted digits, since the code is read as an octal number; if "55" is entered, it will become "055", not "550".

Figure B-3 shows the screen input panel for the feature color table.

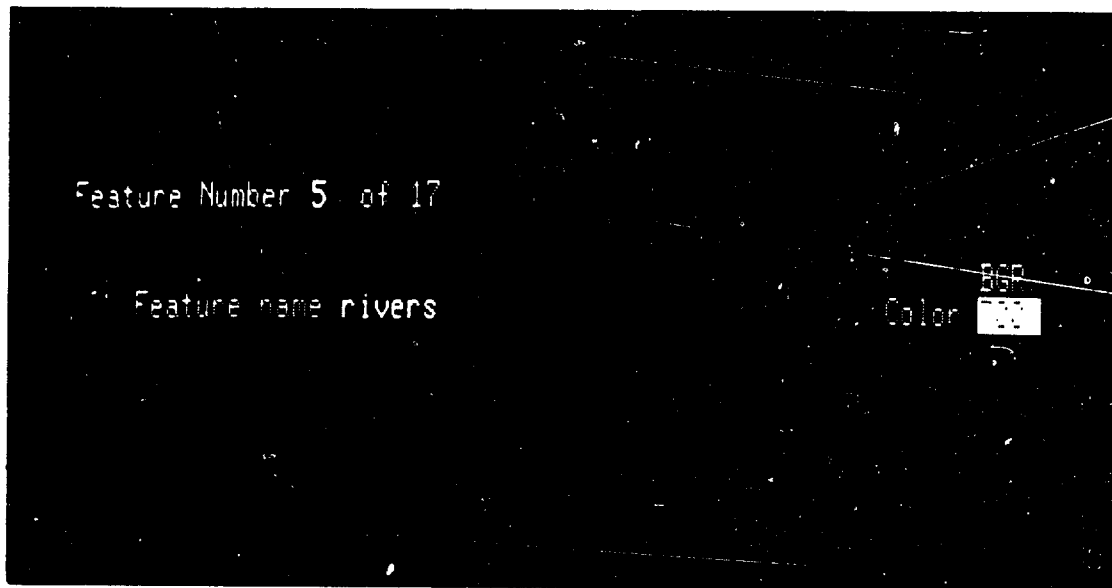


Figure B-3

#### B.4.3 Feature Intersections

After the last feature's name and color have been entered, the screen is formatted with a panel for specifying feature intersections. If there are no intersections in the application,

just press the RETURN key without entering any data. Otherwise, an intersection color, a flag indicating a "strong" or "weak" intersection, and a list of feature names must be entered.

B.4.3.1 Intersection Color. Enter a three digit color code, as for normal feature colors.

B.4.3.2 Strong or Weak. A strong intersection results in the intersection color being used if the feature intersection is present with or without any other features. A weak intersection will only prevail if the exact feature list is present, with no others. Enter "strong" or "weak" to select the proper type.

B.4.3.3 Feature List. Type the feature names which comprise the intersection specification, separated by commas. If there are too many to fit on the line, stop between names (type the last comma) and press RETURN; the next line will be activated as a window for additional feature names.

Figure B-4 illustrates the entry of a feature intersection specification.

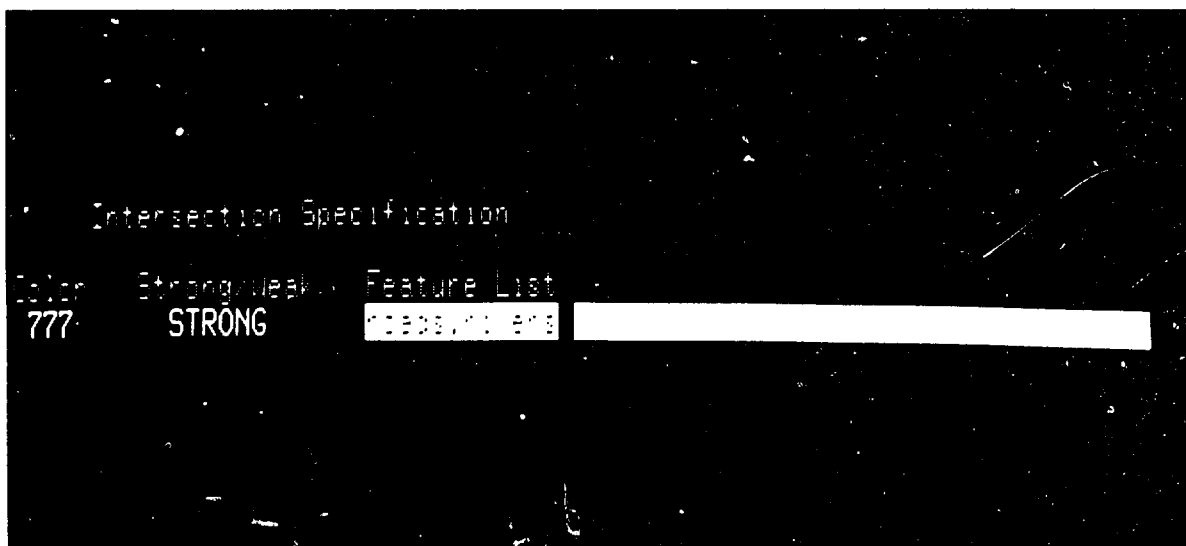


Figure B-4

## B.5 DEFINING THE APPLICATION DEFINITION FILE

The Application Definition File contains parameters used in actually loading the map images from the video disc. It is created through a series of screen input templates, for each map level.

### B.5.1 Intensity Mapping Information

Intensity mapping memory can hold four tables for each color group. Currently, functions are loaded to perform gamma correction, inverse gamma correction, inverse linear correction, and a linear function (i.e., no correction). For each color, the desired function must be indicated; this is done by entering "G", "-G", "-L", or "L", respectively. The name of the file containing the data to be loaded into intensity mapping memory must also be specified; currently, this file is "dmp/intmem".

### B.5.2 Background Control

There are three possible allocations of VFS bitplanes between background and foreground frames: 12/12, 15/9, and 18/6. Enter 0, 1, or 2, respectively, to select the appropriate allocation.

At this stage, the background mode is chosen as black and white or color. (Section 4.2 of the main body of this paper describes the options available for constructing backgrounds.)

Then, the read masks are assigned for the background planes. One mask must be specified for each color. Since the high order bits are used for the background, only the top 4, 5, or 6 bits may be specified, according to an allocation type of 0, 1, or 2. Table B-1 summarizes the background bit plane restrictions.

background/ foreground allocation	no. of background planes per color	available mask bits	maximum octal mask value
12/12	4	11110000	360
15/9	5	11111000	370
18/6	6	11111100	374

Table B-1

Background Bit Plane Restrictions



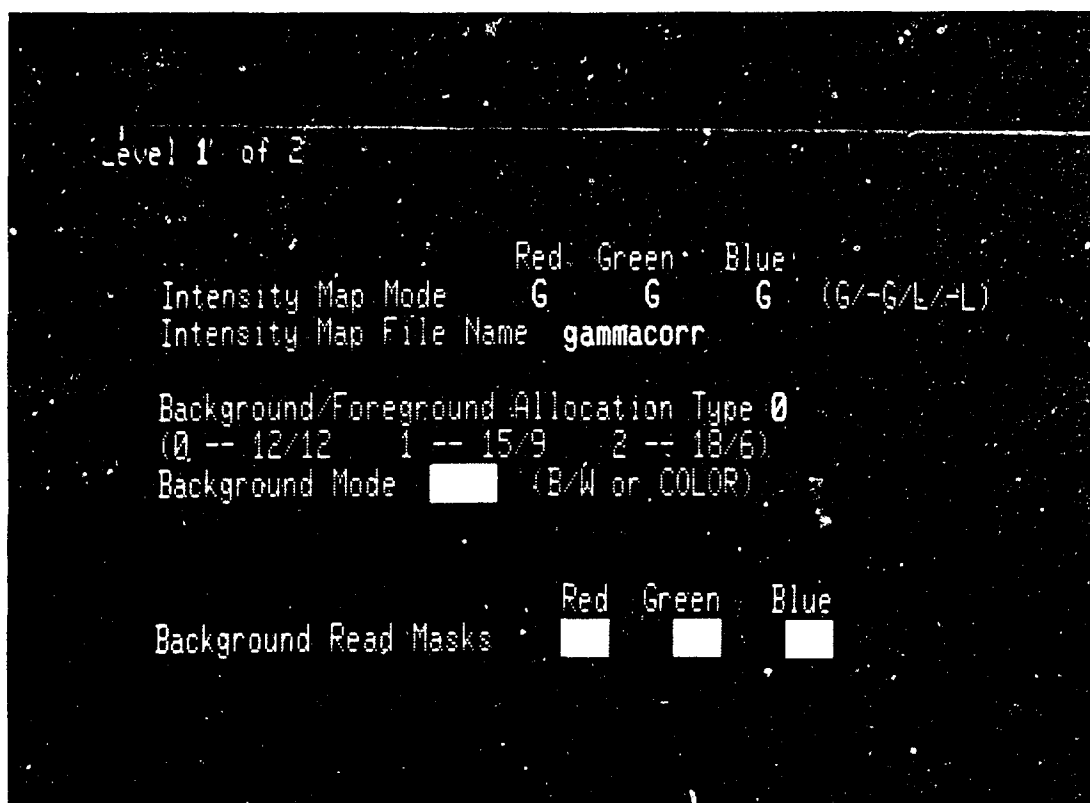


Figure B-5

Figure B-5 shows the Application Definition File panel, with windows for intensity map information and background control.

### B.5.3 Black and White Background Frames

If the background mode was chosen as black and white, certain parameters must be entered for each individual frame. This is done through a separate screen input panel with windows for the following information:

B.5.3.1 Write Masks. The bitplanes of each color group which will receive this frame are specified, with the restrictions outlined in table B-1.

B.5.3.2 A/D Mode. By adjusting the A/D mode (and A/D limits, below) for each individual frame, the optimal digitization can be achieved, with minimum noise and maximum capture of relevant information. Any combination of four A/D capabilities may or may not be activated by typing a "Y" or an "N" next to that item's prompt. The items are: hard limit (accept only pixels with value within indicated limits), saturate (pad pixels outside limits with 1's), and/or (all three color channels must be within limits to produce output), and smear (pixels accepted on the basis of one channel are "smeared" to the other two color groups as well). Smear can only be selected if and/or is not selected.

B.5.3.3 A/D Limits. For each color group, a lower and an upper A/D limit are specified. These limits are used when in the hard limit mode to set the range of acceptable video levels. These values range from 0 through 77 (octal). They are arranged in a 3 by 2 rectangular array of windows which are entered "simultaneously". The cursor may be maneuvered left, right, up, and down among the windows by using the keys F11, F12, F15, and F16, respectively. When the RETURN key is pressed, all six values are accepted.

Figure B-6 shows the input panel for background frames, while the A/D limits are being entered.

#### B.5.4 Foreground Frames

The final information needed at each level associates A/D settings with the actual foreground feature frames present. A screen panel is set up for foreground frame information, with windows for the following:

B.5.4.1 Feature Name. Type the name of a feature previously entered in the Feature Color Table. The same feature may not be used twice at the same level; if a feature name is repeated, an error message will be printed.

B.5.4.2 Feature Type. This one-character code indicates whether the feature is (M)andatory (displayed by default), (P)rimary (loaded by default, but not displayed), or (S)econdary (not loaded from disc until it is explicitly requested). The number of mandatory and primary features is restricted to the number of bitplanes available for foreground frames; when all bitplanes have been exhausted, the remaining features are all set to secondary automatically.

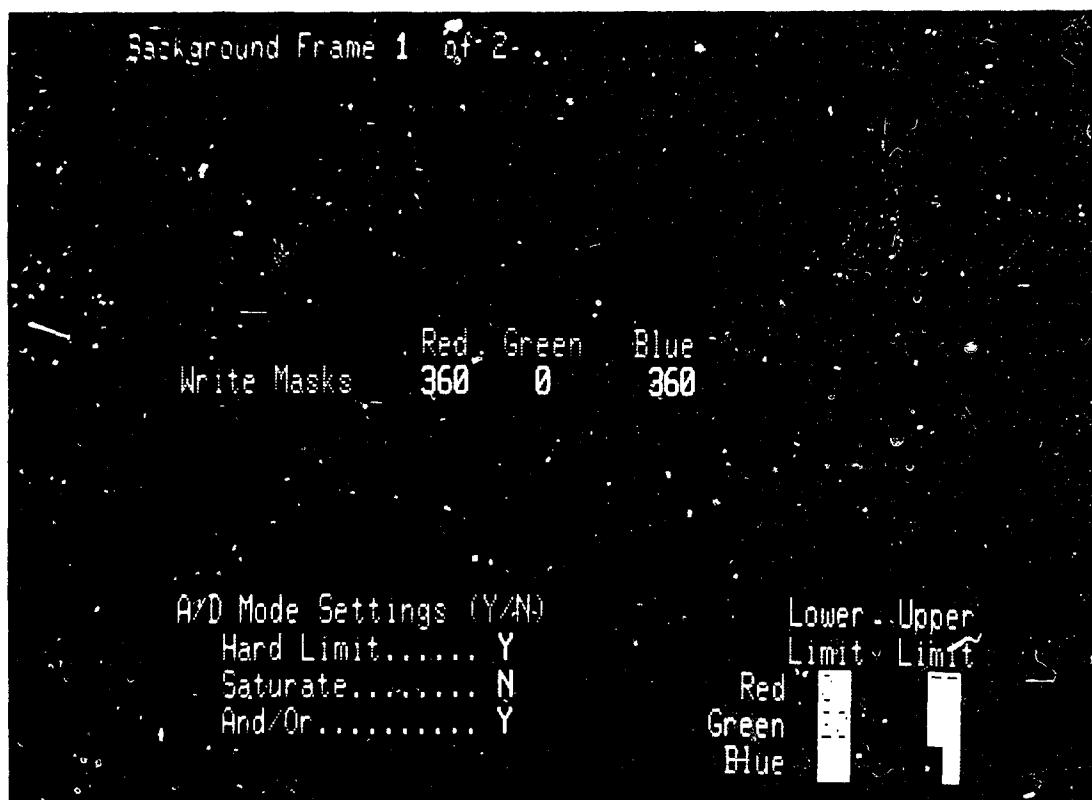


Figure B-6

B.5.4.3 Bitplane Number. The list of available bitplane numbers is displayed, from which one is chosen for this feature. If the feature is secondary, no bitplane is needed, so this item is skipped.

B.5.4.4 A/D Mode. Select from among the four A/D capabilities, as for the black and white background frames, paragraph B.5.3.2 above.

B.5.4.5 A/D Limits. Set the lower and upper limits for the red, green, and blue A/D convertors, as for the black and white background frames, paragraph B.5.3.3 above.

Figures B-7 and B-8 illustrate the screen input panel at two different stages of foreground frame specification at a particular level.

```

Foreground Frame 4 of 14

Feature Name lakes

Feature Type M ('M', 'P', or 'S')

Bitplane Number 8 (3,4,6,7,8,9,10,11,12)


A/D Mode Settings (Y/N)
Hard Limit..... ☒
Saturate..... ☒
And/Or..... ☒

Lower Upper
Limit Limit
Red ☒ ☒
Green ☒ ☒
Blue ☒ ☒

```

Figure B-7

After all information requested in the foreground frame panel has been provided, the screen is cleared and the original Application Definition File panel is put back on the screen, for input of the next level's information.

#### B.6 DATA ENTRY ERRORS

If you make an error while entering data, and press the RETURN key before you notice it, try to finish the data entry cycle with correct data. (This may not be possible, for example, if an incorrect number of levels or number of features is specified.) After the last foreground frame information for the last level has been fully specified, the final screen panel is removed. The name

of the editable file is displayed on the screen and the definition program exits. Before proceeding to pass 2, invoke the text editor and make the necessary corrections to the data.

If an error occurs during pass 2, the error message will be printed on the screen, along with the line number in the editable file where the error was detected. This could only happen if a change was made to the file after pass 1 completion. The line number and error message should make it possible to syntactically and semantically restore the editable file.

```

Foreground Frame 13 of 14

Feature Name contours
Feature Type S ('M', 'P', or 'S')
Bitplane Number 0 (ALL BITPLANES HAVE BEEN ALLOCATED)

A/D Mode Settings (Y/N)
Hard Limit..... Y
Saturate..... N
And/Or..... N
Smear..... N

Lower Upper
Limit Limit
Red
Green
Blue

```

Figure B-8

## APPENDIX C

### APPLICATION CONTROLLER USER'S MANUAL

#### C.1 INTRODUCTION

The Application Controller is an interactive program designed to test and demonstrate the use of the VFS and its terrain map display system software.

Users of the Application Controller who are running VFS demos are VFS project members, familiar with the VFS architecture and map database format. It is for these users that this manual is intended.

#### C.2 BASICS

The Application Controller program resides in the directory "vfs"; the easiest way to invoke it is to login under the "vfs" account. The program name is "application" and is invoked with a unique application name as its first argument. (Additional arguments depend on the mode in which the program is run, described below.)

To activate a map display application, the application database must already have been defined. This is done with the Application Definition package.

##### C.2.1 Using the Video Disc

The standard way to use the Application Controller is with a video disc on which the map images are stored. In this mode, no further arguments other than the application name are required, and the program is invoked as in the following example:

```
Zapplication fuldademo
```

##### C.2.2 Using a Pre-Loaded Image

In some cases, it is not desirable to use the video disc as the source for map images. For example, a special demonstration may be prepared which involves a geographic region not included on the demonstration disc. In this case, the Application Controller is run in a special mode in which the disc is never accessed for new map

images. This requires that the user pre-load the image (using commands in the VFS Executive) before running the Application Controller. The Controller is invoked with a special second argument to indicate suppression of video disc accesses:

`%application fuldademo -`

### C.2.3 Loading a Digital Image File

An extension of the no-disc-access mode includes the loading of a previously dumped image file. The map image would have to have been prepared in advance under the VFS Executive and dumped to a file. When the Controller is run in this mode, the file is loaded into frame store memory, and no disc accesses are performed throughout the execution of the program. A second argument is required, giving the name of the file:

`%application fuldademo /data3/fuldademo`

In both modes which ignore the video disc, an application database is still necessary to provide feature colors and bitplane numbers. It is therefore important that the features be loaded into the bitplanes indicated in the database, or effect of the feature selection and color control commands will be unpredictable.

## C.3 COMMAND SYNTAX

Application controller commands obey a uniform syntax. A command line consists of the command name followed by any arguments which the command requires. The delimiter between argument and between the command name and the first argument may be a space or a comma. It is natural to use a space after the command name and commas between arguments, e.g.,

`>add roads,rivers,cities`

## C.4 INDIRECT COMMAND FILES

The application controller supports the use of indirect command files. This feature is particularly handy in running "canned" demos. The Application Controller may be directed to accept input from an indirect command file by typing the character "<" followed by the file name.

Comments may be inserted in an indirect command file by typing a semicolon (";") as the first character of the line. When this line is encountered during execution, it will be echoed to the terminal screen, providing a facility for status messages and prompts.

## C.5 UNIX COMMAND EXECUTION

Any UNIX command may be executed while running the Application Controller by preceding the command with the character "!".

## C.6 COMMANDS

Table C-1 lists the commands for quick reference, indicating the arguments expected and giving a brief description. The commands are described in detail in this section, grouped according to the type of command.

<u>Command</u>	<u>Argument(s)</u>	<u>Description</u>
add	list of features	enable feature display
erase	list of features	disable feature display
remove	list of features	disable feature display and free bitplanes
color	feature, color	set feature color
color	color, strong/weak, list of features	set intersection color
precede	feature, feature	change feature precedence
zoom	factor	zoom the display
translate	line #, pixel #	translate the display
slide	# lines, # pixels	slide the display
joystick	key displacement	simulate joystick device
disc	on or off	display video disc image
search	frame #	search video disc
query	---	environment information
load	file name	load a dump file
sync	---	restore synch
pause	---	await keyboard response

Table C-1

### Application Controller Commands



### C.6.1 Feature Selection

Features may be selected for display or removed from the screen and are identified by feature name. Abbreviations may be used, and an appropriate message is printed if too few letters were specified to disambiguate the feature name.

C.6.1.1 ADD. The ADD command takes a list of feature names and enables the bitplanes in which they have been loaded. Secondary features are loaded from the video disc into available bitplanes. If all bitplanes are in use, a message is printed on the screen; the REMOVE command must be used to free a bitplane for the new feature (see below).

C.6.1.2 ERASE. The ERASE command takes a list of feature names and erases them from the screen. If other features were obscured by the features being removed, the "hidden" pixels will show the obscured information upon erasure. The features erased remain in the bitplanes for immediate selection at a later time. (That is, ERASE changes feature types from mandatory to primary.)

C.6.1.3 REMOVE. The REMOVE command is a stronger form of the ERASE command. The features listed are erased from the screen, and their bitplanes are freed for use by other features not yet loaded from the video disc.

### C.6.2 Color Control

Color of individual features, relative feature display precedence, and intersection coloring can all be accomplished by simple commands.

C.6.2.1 COLOR. The COLOR command has two arguments: the first is a feature name, and the second is a three digit octal code indicating intensity values of blue, green, and red (left to right). Each digit may be between 0 and 7. For example:

```
>color rivers 660
```

will set the feature "rivers" to a turquoise color.

C.6.2.2 COLOR for Intersections. Intersection colors are specified using the same command as for feature colors. The color of previously defined intersections may be changed, or new intersections may be specified. When using the COLOR command for intersections, the first argument is the three digit color code. The second argument indicates whether the intersection is strong or weak. If an intersection is weak, the intersection color will only

be used if precisely the features listed for the intersection are present. Strong intersections force the intersection color even when additional features are present. If neither "strong" or "weak" is specified, "weak" is assumed. After the optional strong/weak argument, the features comprising the intersection are listed. For example,

>color 0 roads,rivers

will color all pixels containing only the features "roads" and "rivers" black;

>color 777 strong highways,railroads

will color all pixels containing the features "highways", "railroads", and anything else white.

C.6.2.3 PRECED. The PRECED command controls relative feature precedence. If one feature obscures another on the display, the user can raise the precedence of the obscured feature to make it visible. The command

>precede rivers,boundaries

will adjust the precedence of all features so that "rivers" has precedence just higher than "boundaries." The command

>precede rivers

will give "rivers" a precedence higher than every other feature.

### C.6.3 Picture Manipulation

The displayed image may be zoomed (magnified) as well as translated in a variety of ways.

C.6.3.1 ZOOM. The ZOOM command accepts a single argument, the zoom factor. If an unsupported zoom factor is entered, an appropriate message will be printed. At this writing, the available zoom factors are 1.25, 1.5, 2, and 4, although more will be added shortly. To return to an unzoomed display, enter a zoom factor of 1.

C.6.3.2 TRANSLATE. The TRANSLATE command shifts the display to a particular line and pixel offset. The arguments are a pair of signed numbers indicating the line offset (positive: up, negative:

down) and the pixel offset (positive: left, negative: right). Areas outside the map image area which appear on the screen as a result of the translation operation are blacked out.

C.6.3.3 SLIDE. The SLIDE command performs a smooth, relative translation. The arguments are the number of lines (positive or negative) and the number of pixels (positive or negative); the screen image will be translated in a continuous, smooth manner the number of lines and pixels indicated.

C.6.3.4 JOYSTICK. This command uses the numeric keypad on the right portion of the keyboard to simulate the action of a joystick input device. The "5" key has no effect; each of the eight keys around the "5" causes the screen image to move in the direction corresponding to the key's location relative to the "5" (e.g., "7" is straight up, "1" is down and left, etc.). With each keystroke, the screen is displaced a single line or pixel, unless an argument is used with the JOYSTICK command. The argument indicates the pixel displacement to be used with each keystroke depression. Using an argument of 3 or 5 and holding the numeric keys down to activate the typamatic action of the keys is particularly effective in simulating a joystick device. To terminate the JOYSTICK command and return to normal Application Control command input, press the "0" key.

#### C.6.4 Video Disc Control

An interface is provided to temporarily switch to a direct display of the image coming from the video disc, without destroying the frame store memory. The disc can also be searched for a particular frame.

C.6.4.1 DISC. The disc command takes a single argument, "on" or "off". The default is "on". "On" causes the image from the video disc to be displayed on the screen; "off" returns to the stored map. The disc display feature permits the integration of disc-based "viewgraph" frames with the VFS demonstration to permit a full briefing to be given from within the Application Controller.

C.6.4.2 SEARCH. The SEARCH command takes a frame number as its argument, and instructs the video disc to search for that frame.

#### C.6.5 Miscellaneous

These four commands provide miscellaneous utility functions.

C.6.5.1 QUERY. The QUERY command lists all features, their colors, and indicates whether the feature is currently displayed, is loaded into a bitplane but not displayed, or was never loaded. If

the image is zoomed, the zoom factor is reported, and if it is translated, the absolute line and pixel number are displayed.

C.6.5.2 LOAD. Any VFS "DUMP-format" file may be loaded by using the LOAD command with the file name as an argument. The most frequently loaded files are dumps of coefficient memory to illustrate different zooming algorithms.

C.6.5.3 SYNC. This command re-synchronizes the microsequencer with the raster scan.

C.6.5.4 PAUSE. The PAUSE command is used in indirect command files and causes execution to halt temporarily until the user presses the RETURN key at the terminal.

## DISTRIBUTION LIST

### INTERNAL

#### D-10

E. L. Key

#### D-11

J. J. Croke

#### D-30

V. A. DeMarines

J. H. Phillips

#### D-33

J. P. Bean

E. D. Bell

J. S. Bigelow

M. J. Bloom

L. A. Chapman

J. T. Connolly

L. Elias

R. Platcow

#### D-34

D. W. Craig

P. J. Cunningham

G. C. Dimock

T. E. Everett

B. W. Fam

S. C. Hemminger

G. L. Hollis

K. Ilse (10)

T. A. James

C. A. Joseph

J. H. Keating

D. H. Lehman (5)

J. K. Millen

C. J. Murphy

J. P. Otin

D. C. Pasterchik

R. A. Porter

W. D. Ricker

M. E. Rudell

J. E. Sachs

J. H. Sangster

B. P. Schanning

I. R. Smith

D. R. Sullivan

G. Tenuta

A. Ward

J. C. C. White

J. P. L. Woodward

#### D-35

R. C. Josephson

#### D-36

S. R. Ames

D. L. Baldauf

D. L. Martell

F. R. Murphy

R. D. Rhode

C. M. Sheehan

#### D-76

H. A. Bayard

### EXTERNAL

#### RADC

F. Rahrig



## *MISSION of Rome Air Development Center*

*RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C<sup>3</sup>I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.*